# More Reliable Binding Affinity Prediction of Protein Ligands Combining Molecular Dynamics Simulations and Machine Learning Models

Master's thesis in Complex Adaptive Systems

MARCUS HANSEN

Department of Physics

CHALMERS UNIVERSITY OF TECHNOLOGY
Gothenburg, Sweden 2025
www.chalmers.se

Master's thesis 2025

# More Reliable Binding Affinity Prediction of Protein Ligands Combining Molecular Dynamics Simulations and Machine Learning Models

MARCUS HANSEN


CHALMERS
UNIVERSITY OF TECHNOLOGY

Department of Physics
Chalmers University of Technology
Gothenburg, Sweden 2025

More Reliable Binding Affinity Prediction of Protein Ligands Combining Molecular Dynamics Simulations and Machine Learning Models
Marcus Hansen



Supervisor: Marco Klähn, AstraZeneca
Examiner: Paul Erhart, Department of Physics

Master's Thesis 2025
Department of Physics
Chalmers University of Technology
SE-412 96 Gothenburg
Telephone +46 31 772 1000

Cover: Protein-ligand complex.

Typeset in LaTeX
Printed by Chalmers Reproservice
Gothenburg, Sweden 2025

More Reliable Binding Affinity Prediction of Protein Ligands Combining Molecular Dynamics Simulations and Machine Learning Models
Marcus Hansen
Department of Physics
Chalmers University of Technology

# Abstract

Efficient and accurate prediction of protein-ligand binding affinities is essential for advancing drug discovery. This thesis presents a novel approach that combines molecular docking results with features derived from short molecular dynamics (MD) simulations to enhance the prediction accuracy of binding affinities. The thesis aims to bridge the efficiency of traditional docking methods with the more accurate Free Energy perturbation (FEP) techniques. MD simulations were conducted on a dataset of around 5,000 protein-ligand complexes and extracted features related to binding enthalpy and dynamic behavior related to the more complex binding entropy. These features, alongside molecular docking scores, were used to train machine learning models. The CatBoost regressor was identified as the most effective model, achieving better predictive accuracy than Molecular Docking alone. This method facilitates more reliable binding affinity predictions by successfully integrating docking insights with dynamic protein-ligand behavior, thereby accelerating the early stages of drug discovery.

Keywords: Protein-Ligand Binding, Binding Affinity Prediction, Molecular Docking, Molecular Dynamics (MD), Machine Learning, Binding Enthalpy, Binding Entropy, Drug Discovery, Free Energy Perturbation (FEP).

# Acknowledgements

I would like to express my deepest appreciation to my supervisor, Marco Klähn at AstraZeneca, for his invaluable guidance and support throughout the development of this thesis. Coming from a different academic background, stepping into the field of chemistry and molecular dynamics simulation presented a big challenge. Marco not only introduced me to this entirely new domain but also consistently encouraged me, patiently addressed all my questions, and generously shared his extensive expertise with remarkable clarity. His mentorship was instrumental to both the progress and successful completion of this work. I am also sincerely grateful to my examiner, Paul Erhart, for kindly accepting the role as examiner and for his time and effort in evaluating this thesis.

Marcus Hansen, Gothenburg, June 2025

# List of Acronyms

Below is the list of acronyms that have been used throughout this thesis listed in alphabetical order:

| | |
|---|---|
| FEP | Free Energy Perturbation |
| fs | Femtosecond |
| K | Kelvin |
| MD | Molecular Dynamics |
| ML | Machine Learning |
| nm | Nanometer |
| ns | Nanosecond |
| PL | Protein–Ligand |
| ps | Picosecond |
| RMSE | Root Mean Squared Error |
| RMSF | Root Mean Square Fluctuation |
| SASA | Solvent Accessible Surface Area |
| SVM | Support Vector Machine |

# Contents

# 1

# Introduction

Discovering new drugs is a complex, multi-step process that is both time-consuming and expensive [1], [2]. It can take over a decade and cost billions of dollars to develop a potential drug, often with no guarantee that it will ever reach the market. The early stages of drug discovery typically begin with identifying a target, such as a protein associated with a particular disease [1]. The goal is then to find molecules (e.g., ligands) that can change the function of this target. This involves screening large databases of ligands to identify those that bind to the target with sufficient binding affinity, that is, how tightly a ligand binds to its protein target [3], to be considered a potential hit finding [4]. While binding affinity is an essential criterion all drugs share, the ideal ligand must exhibit a favorable overall property profile [5]. This includes acceptable toxicity properties and desirable ADME (absorption, distribution, metabolism, and excretion) properties. The next stage, lead optimization, focuses on refining these molecules while maintaining certain parts of the molecule (the scaffold) unchanged to improve their effectiveness and develop drug candidates with favorable properties.

Traditionally, these early steps in drug discovery were carried out experimentally, making the process slow and costly [4], [6]. Computational techniques have been introduced to accelerate and guide the early stages of drug discovery, helping to prioritize candidates for experimental validation rather than replace laboratory testing. One of the most widely used approaches is molecular docking [6]. Molecular docking enables virtual screening by predicting how a ligand fits into a target site and estimating the binding affinity, using an empirical potential to describe protein-ligand interactions [4], [7]. This method provides a faster and more cost-effective way to filter out weak binders and prioritize promising candidates for further study. While docking has helped in narrowing down potential ligands by filtering out likely weak binders, it comes with limitations. The high computational speed often comes at the expense of accuracy, as docking relies on simplified models that, in the best case, only capture some of the dynamic nature of molecular interactions [6], [7].

A smaller subset of candidate ligands is retained for further analysis following molecular docking. At this stage, more accurate and computationally intensive methods become feasible [8]. Techniques based on molecular dynamics (MD) simulations and statistical mechanics, such as free energy perturbation (FEP), have demonstrated high accuracy in predicting binding affinities [9]. Unlike docking, FEP uses exten-

sive MD simulations to model the physical interactions between ligands and proteins in greater detail. While FEP is considered more robust and reliable, it is also extremely time-consuming and resource-intensive, making it suitable primarily for the lead optimization phase, where fewer molecules remain to be evaluated [4], [8].

This project proposes a novel approach to bridge the gap between the efficiency of molecular docking and the accuracy of FEP methods. Integrating information derived from molecular docking with short MD simulations aims to capture essential dynamic aspects of protein-ligand interactions often overlooked by docking. A machine learning model will be trained to predict binding affinity, using features extracted from MD simulations and initial docking scores as input. Experimentally determined binding affinities will serve as the target values. The primary goal is to develop a computational method that offers a significantly more efficient alternative to FEP while achieving higher predictive accuracy than docking alone, thereby enabling the screening of a larger number of ligands with greater accuracy.

## 1.1 Aim

This project aims to develop a fast and accurate method for predicting protein-ligand binding affinities with performance comparable to FEP. This goal will be pursued through four main tasks:

1. Performing short MD simulations of protein-ligand complexes, starting from experimental x-ray structures.
2. Extracting features related to binding enthalpy and dynamic behavior related to the more complex binding entropy.
3. Developing a machine learning (ML) model capable of predicting binding affinity using the extracted features and docking scores.
4. Evaluating the developed model against experimentally measured binding affinities and predictions made by FEP.

Beyond these primary tasks, the thesis will also explore the importance of features in identifying which ones are most predictive of binding affinity and which may be redundant. This provides insight into how molecular properties influence binding affinity.

# 2

# Theory

## 2.1 Protein-Ligand Interaction

Binding affinity prediction is critical in drug discovery, particularly during its early stages [10]. Finding a ligand that binds strongly to its target protein is challenging and costly, but it forms the foundation for much of a drug's activity [11]. Accurate estimation of binding affinities of drug candidates helps accelerate drug development by filtering out weak binders and prioritizing strong candidates [12]. Due to its impact on the efficiency and success of new drugs, binding affinity remains a significant focus in the field.

This section will describe the fundamental principles of protein-ligand interactions, including the factors and forces that drive binding kinetics and the relationship between entropy and enthalpy and these kinetics.

### 2.1.1 Binding Kinetics

The first mechanism to be introduced is protein-ligand binding kinetics [4]. This describes the rate and efficiency of protein-ligand association (binding) and dissociation (unbinding), which is illustrated in Figure 2.1.

P represents the protein, L is the ligand, and PL is the protein-ligand complex formed through their association in a solution, such as water. The rate constant $k_{on}$

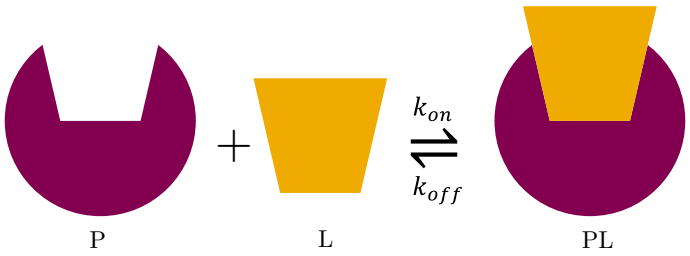



**Figure 2.1:** Binding between a protein (P) and a ligand (L) forming a protein–ligand (PL) complex. The association rate is denoted by $k_{\text{on}}$, and the dissociation rate by $k_{\text{off}}$.

describes the association rate, while $k_{off}$ describes the dissociation rate.

At equilibrium, the rate of association equals the rate of dissociation. This can be expressed as:

$$k_{on}[\mathrm{P}][\mathrm{L}] = k_{off}[\mathrm{PL}] \tag{2.1}$$

From this equilibrium expression, the binding constant $K_b$ can be introduced:

$$K_b = \frac{k_{on}}{k_{off}} = \frac{1}{K_d} \tag{2.2}$$

where $K_d$ is the dissociation constant and is inversely related to binding affinity. A low $K_d$ indicates stronger binding, and a fast association rate (high $k_{on}$) combined with a slow dissociation rate (low $k_{off}$) favors the formation of a stable PL complex [4].

The binding affinity between a protein and a ligand is commonly expressed as the dissociation constant ($K_d$) or as the binding constant ($K_b$), or their logarithmic forms ($pK_d$ or $pK_b$, defined as $-\log(K_d)$ and $-\log(K_b)$ respectively) [13]. In this thesis, $pK_b$ is used to measure binding affinity, as it aligns with the format in which the experimental values in the dataset were reported.

### 2.1.2 Thermodynamic Principles of Protein-Ligand Binding

The next mechanism to be discussed is the thermodynamic principles of binding. A PL-solvent system consists of the protein, the ligand, and the surrounding solvent (water and ions) [4]. The interactions are not limited to the protein and ligand alone but also involve the solvent, making it a large and complex system that follows the laws of thermodynamics.

The most important thermodynamic quantity for binding is the Gibbs free energy ($\Delta G$) [4]. This quantity describes the favorability of a binding event. It is measured by the energy difference between the ligand and protein separated in water (left side of figure 2.1) and protein and ligand in water during binding (right side of figure 2.1). Spontaneous binding occurs when $\Delta G$ is negative. A more negative value indicates a more stable PL complex, corresponding to stronger binding between the ligand and the protein.

The binding constant ($K_b$) introduced in Section 2.1.1 can be related to thermodynamics by defying the standard binding free energy ($\Delta G^\circ$), which represents the Gibbs free energy change under standard conditions (1 atm pressure, a temperature of 298 (Kelvin) K, and a PL concentration of 1 M) [4]. The relationship is given by:

$$\Delta G^\circ = -RT \ln K_b \tag{2.3}$$

where R is the gas constant, and T is the temperature. This equation shows that $K_b$, which describes the binding kinetics, influences the stability of the complex and the binding strength between the protein and ligand.

The binding free energy at any point during the interaction is given by:

$$\Delta G = \Delta G^{\circ} + RT \ln Q \tag{2.4}$$

$Q$ reflects the ratio of the PL complex to the free protein and ligand concentrations. At equilibrium, $Q = K_b$ and $\Delta G = 0$.

Gibbs free energy can also be expressed in terms of enthalpy and entropy changes before and after binding, which will be touched upon multiple times in the thesis:

$$\Delta G = \Delta H - T\Delta S \tag{2.5}$$

The enthalpy change ($\Delta H$) includes the energy gained from forming non-covalent interactions, such as hydrogen bonds, van der Waals forces, and electrostatic interactions, between the protein and ligand, as well as the energy required to break interactions with surrounding water molecules and reorganization of the water when the protein binds [4]. Another component the enthalpy change captures is the reorganization energies of the protein when going from a non-binding to a binding state. It thus captures both favorable and unfavorable energetic aspects of the binding process.

The entropy change ($\Delta S$) reflects the system's disorder. Binding typically reduces the translational and rotational freedom of the protein and ligand (unfavorable). However, it can increase entropy by releasing water molecules from the binding site into the solvent (favorable) [4]. The total entropy change combines solvent reorganization, molecular flexibility changes, and molecular motion loss. The total entropy change can be decomposed into the following components:

$$\Delta S = \Delta S_{\text{solvation}} + \Delta S_{\text{conformational}} + \Delta S_{\text{rotational/translational}} \tag{2.6}$$

For binding to occur, strong interactions or solvent entropy gain must compensate for the unfavorable entropy changes.

The entropy change depends heavily on how the mobility of the ligand and protein changes upon binding [4]. To accurately estimate binding affinity, this change in entropy must be captured. FEP achieves this by simulating both the bound (right side of figure 2.1) and unbound (left side of figure 2.1) states using MD, thereby accounting for these entropy differences, a key reason for their predictive accuracy [9]. Other methods for estimating this entropy change also exist, such as normal-mode analysis, which estimates it by examining how the vibrations of the ligand change upon binding [14]. However, they are not that reliable.

As mentioned, PL binding is driven by a decrease in Gibbs free energy ($\Delta G$), which depends on entropic and enthalpic changes. Conversely, favorable entropy gains often involve enthalpic penalties. This balance is known as enthalpy–entropy compensation, where opposing changes in enthalpy ($\Delta H$) and entropy ($\Delta S$) result in changes in $\Delta G$ [4]. This compensation likely arises from alterations in weak non-covalent interactions and is influenced by factors such as solvation, ligand structure, binding site flexibility, and water dynamics.

## 2.2 Computational Techniques of Binding Affinity Predictions

Having established the basics of PL interactions, this section introduces computational techniques for estimating binding affinity, including molecular docking and MD simulations.

Since the aim of this thesis is to build upon molecular docking by combining its information with insights from MD simulations to predict binding affinity using an ML model, it is crucial to understand both methods. Section 2.2.1 will describe the principles of molecular docking and the types of information it provides. Section 2.2.2 will cover MD simulations and provide an overview of their methodology, which is described in more detail in Section 2.2.3.

### 2.2.1 Molecular Docking

Molecular docking is a computational technique commonly used in the early stages of drug discovery to predict how a ligand (potential drug candidate) interacts with a protein (target) [7]. Docking software typically consists of two main components: a pose generation step that identifies possible binding modes between the ligand and protein by translating and rotating the ligand close to the protein active site, and then the ligand groups are rotated to find an optimal fit inside the protein active site and a scoring function that evaluates and ranks each pose based on estimated intermolecular interactions (e.g., hydrogen bonds, hydrophobic contacts), which are mainly enthalpic, to approximate the binding affinity. This is illustrated in a simplified way in Figure 2.2, where different ligand poses are evaluated by docking them to the protein. The pose with the best fit is ranked highest (e.g., rank 1), while less favorable poses receive lower rankings (e.g., rank 3).

Molecular docking is an efficient tool for screening large ligand libraries, helping filter out weak binders and prioritize strong candidates [7]. To achieve the necessary computational efficiency, docking methods introduce certain simplifications, such as treating the protein as rigid, with only limited flexibility allowed for a few residues [7]. These approximations can affect the accuracy of the predicted poses and, consequently, the binding affinity estimates.

With this understanding of molecular docking, the next step is to introduce the software used to estimate the binding affinity of the PL complexes in this thesis. GNINA [15], a molecular docking tool, was used to provide an initial computational prediction of binding affinities. These affinities serve as part of the input to the ML model developed in this work.

GNINA [15] extends traditional docking frameworks such as AutoDock Vina [16] and Smina [17] by incorporating convolutional neural networks (CNNs) to re-rank ligand poses. This ML-enhanced scoring function aims to improve the accuracy of

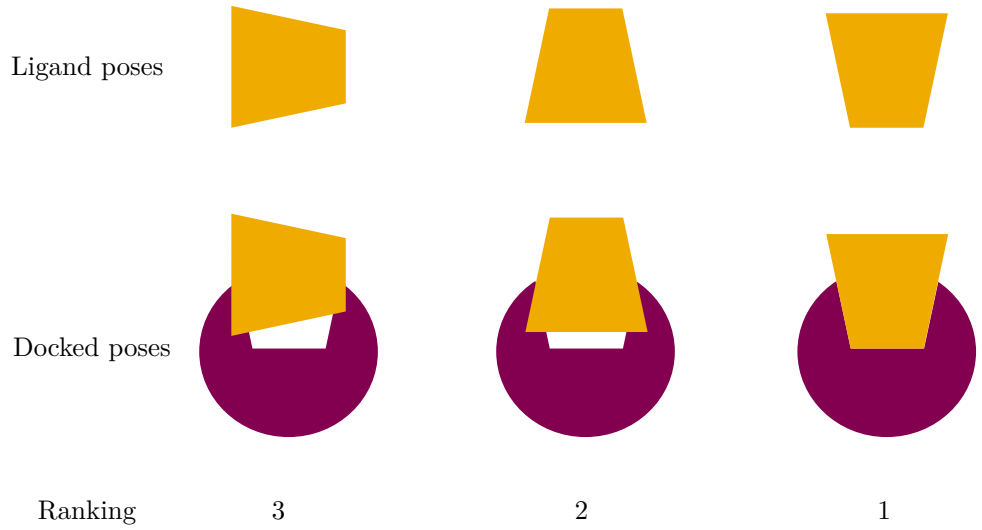



**Figure 2.2:** Simplified illustration of the molecular docking process. Different ligand poses are docked to the protein target. The best-fitting pose receives the highest rank.

binding affinity predictions over classical scoring approaches.

### 2.2.2 Molecular Dynamic Simulations

MD simulations provide a detailed view into the dynamics of protein behavior and their interactions with other molecules like, for example, ligands [18], [19]. Observing these interactions over time helps us understand protein function and how ligand binding can change a protein's structure, dynamics, and conformation. MD simulation gives us a way of understanding the dynamics of the entire PL complex, which can help us in determining the stability [19].

MD simulations are a powerful tool for studying the motions of PL complexes at an atomic level. MD simulations calculate the forces acting on each atom from all other atoms in the system based on their positions. Newton's laws of motion use these forces and current velocities to update the positions and velocities over time [20]. Repeating this process across many time steps produces a trajectory that captures the system's dynamic behavior.

Since each atom experiences forces from all other atoms, resulting in $N^2$ pairwise interactions for a system of $N$ atoms [21]. In practice, the pairwise interactions are only calculated between the atoms within a certain distance, and everything beyond that is calculated in Fourier space using Ewald sum-based methods. MD is computationally demanding with the need for very short time steps of 1 femtoseconds (fs) to capture the system's fastest motion, which are the atoms bonded to the light hydrogen atoms and simulations often spanning nanoseconds (ns) or longer [21].

The forces between atoms are described using force fields, which are computational

models that define a system's potential energy as a function of atomic positions [4], [21]. The forces are then calculated as the negative gradient of this potential. Force fields account for both bonded interactions, such as bond stretching, angle bending, and torsions (the first four terms in Figure 2.3), and non-bonded interactions, such as van der Waals forces and electrostatics (the last two terms in Figure 2.3). This enables an approximate representation of the physical behavior of molecular systems [18].

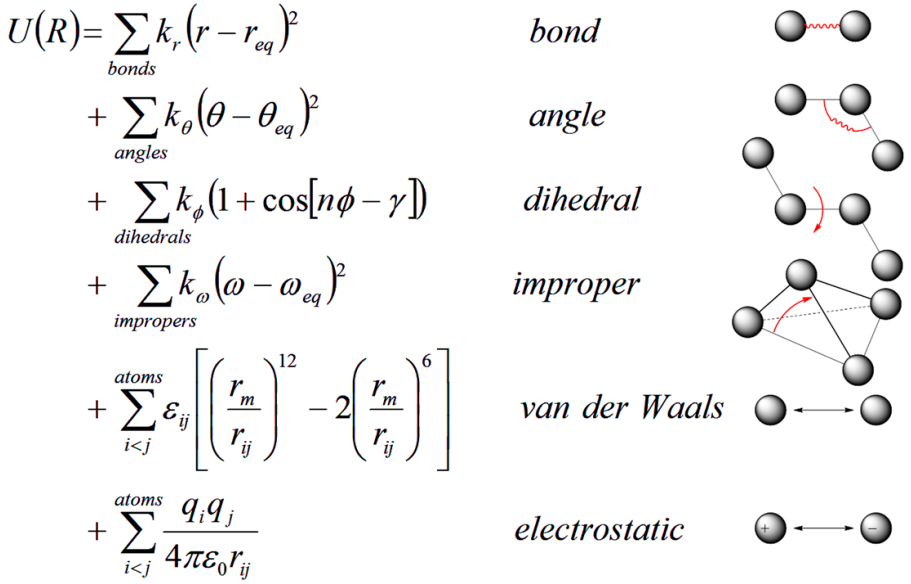



**Figure 2.3:** The force field is described by the different interaction types shown on the right and their corresponding energy functions on the left. Key parameters include bond length ($r$), angles ($\theta$, $\phi$, $\omega$), interatomic distances ($r_{ij}$, $r_m$), force constants ($k$), equilibrium positions, atomic charges ($q$), and the dielectric constant ($\epsilon_0$) [22], CC BY 4.0.

Once the trajectory has been generated, properties of the simulated system can be estimated by averaging over the simulation frames. This approach is grounded in the ergodic hypothesis from statistical mechanics [23]. The ergodic hypothesis states that, given an infinitely long simulation, every accessible point in phase space will be visited. Therefore, this allows us to approximate the ensemble average, which represents all possible states of the system through time averages over a single trajectory. However, in practice, the lengths of trajectories are finite, leading to potential systematic errors in the estimates.

### 2.2.3 Molecular Dynamic Simulation Steps

Before starting an MD simulation, a starting structure of the PL complex is required, detailed information about the positions and arrangement of all atoms [3]. The protein structure can be obtained experimentally using techniques such as X-ray crystallography or nuclear magnetic resonance spectroscopy [24] or predicted com-

putationally using methods like AlphaFold [25]. The ligand structure can also be determined experimentally or approximated through molecular docking, although docking is generally less accurate than experimental approaches.

### Step 1: Preparing the system

System preparation for MD simulations consists of several steps. The first step is to generate a topology file containing all necessary information about the system, including the protein, ligand, and solvent, as well as the interactions between atoms, based on the selected force field [26]. Topology files are simulation system-specific.

The next step is to solvate the system by placing the PL complex in a simulation box, adding water molecules, and introducing ions to neutralize the system's net charge [26]. Water is added to mimic the biological environment, as simulations in a vacuum are not physiologically relevant.

After solvation, periodic boundary conditions (PBC) are applied [26]. This means that copies of the simulation box are placed around the original. Atoms near the boundaries of the box interact with atoms in neighboring images. If an atom moves out of the box on one side, it re-enters from the opposite side, maintaining a constant number of particles. PBC is used to avoid artificial surface effects and to ensure that atoms at the boundaries still experience a realistic environment [26].

### Step 2: Energy minimization

Energy minimization is important before starting an MD simulation [18]. When a system is first built, atoms may be positioned in ways that create unrealistic or high-energy interactions, such as being placed too close together. To resolve this, energy minimization adjusts atomic positions to reduce the system's potential energy, resulting in a more stable and physically realistic starting structure.

This is achieved using numerical algorithms, such as steepest descent, that iteratively move atoms in the direction that lowers the potential energy, based on the forces calculated from the force field [18]. This process is done until the forces are small, indicating that the system has reached a low-energy state suitable for simulation.

### Step 3: Equilibration

After obtaining a stable starting structure through energy minimization, thermally equilibrating the system is the next important step before running the final simulation. This is necessary because the system is initially far from equilibrium. After all, atomic velocities are assigned randomly, and the water needs time to reorganize around the PL complex [26]. During equilibration, restraints are applied to the protein and ligand, allowing the solvent to reorganize around them appropriately.

Equilibration is typically carried out in two phases. First, an NVT equilibration is

performed to bring the system to the desired temperature, where NVT stands for constant Number of particles, Volume, and Temperature. Secondly, pressure is also controlled to adjust the system's density to a realistic value (NPT: constant Number of particles, Pressure, and Temperature) [26]. This process brings the system closer to experimental conditions.

**Step 4: Production MD**

After equilibration, the system is ready for the main simulation phase, commonly called Production MD [26]. The MD trajectory is generated during this phase at the desired temporal resolution. For example, if a two ns simulation is performed and atomic coordinates are saved every 10 picoseconds (ps), the trajectory will consist of 200 frames. The atomic coordinates must be saved more frequently if higher resolution is desired.

## 2.3 Molecular Dynamic Features

This section will introduce the MD features used in this thesis. As discussed in Section 2.1.2, enthalpic and entropic contributions are key driving forces behind PL binding. Therefore, the aim is to select features that reflect these thermodynamic components, particularly the three entropy terms described in Equation 2.6. In addition to entropy-related features, features capturing enthalpic interactions, ligand structure, and binding site flexibility are also considered. The primary goal of incorporating these features is to support the prediction of binding affinity.

However, emphasis is placed on entropic features because, as mentioned in Section 2.2.1, docking scores primarily capture enthalpic contributions. Since this thesis combines docking scores with MD-derived features, focusing on entropic factors, typically underrepresented in docking-based approaches, is beneficial.

A detailed explanation of how each feature was extracted will be provided in the Methods section 3.3. The following section will provide a brief introduction to each feature, along with the motivation for its selection.

### 2.3.1 Root Mean Square Fluctuations

Root mean square fluctuation (RMSF) measures atomic fluctuations during a simulation [18]. It quantifies the average deviation of a particular atom from its average position throughout the simulation. The following equation can describe the RMSF for an atom $i$:

$$\mathrm{RMSF}_i = \sqrt{\langle (r_i - \langle r_i \rangle)^2 \rangle} \tag{2.7}$$

Where $r$ is the positional vector $(x, y, z)$ and $\langle \cdots \rangle$ denotes the time average over the simulation frames.

As mentioned in [27], RMSF is an important value for understanding the flexibility of a protein's binding site. A highly flexible binding site (indicated by high RMSF) leads to larger entropy, which is always favorable (see equation 2.5). However, when the ligand binds to the protein, the ligand loses flexibility and gets an entropy penalty.

### 2.3.2 Dihedral Angle

Four atoms, forming two planes, define a dihedral angle: one plane formed by atoms A–B–C and another by atoms B–C–D, as illustrated in Figure 2.4. The dihedral angle is the angle between these two planes.

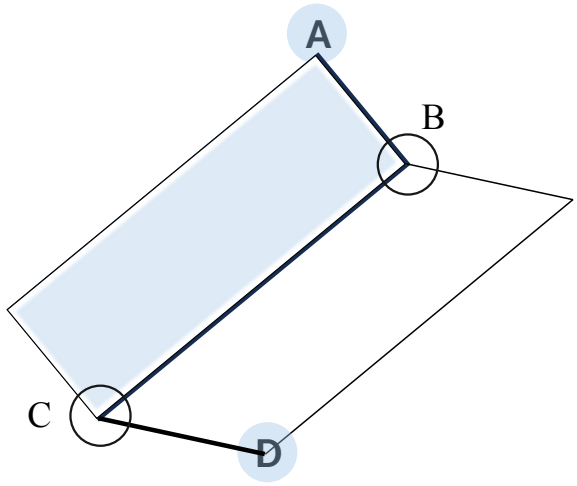



**Figure 2.4:** Dihedral angle defined by four atoms (A–B–C–D). The angle is formed between the plane of atoms A–B–C and the plane of atoms B–C–D.

Dihedral angles are directly related to conformational entropy ($\Delta S_{\text{conformational}}$) as described in Equation 2.6 [28]. Previous studies have also shown that dihedral angles change upon ligand binding [29].

### 2.3.3 Solvent Accessible Surface Area

Solvent Accessible Surface Area (SASA) is a measure that describes the area of a molecule that is accessible to solvent molecules, such as water [30]. The SASA is illustrated in figure 2.5 for a protein before and after ligand binding. When unbound, the entire surface of the protein is accessible to the solvent (blue-shaded region). When the ligand binds, that area of the protein is no longer accessible to the solvent; hence, the SASA value has decreased.

**Figure 2.5:** SASA for a protein before and after ligand binding. When unbound, the entire surface of the protein (blue-shaded area) is accessible to solvent molecules. Upon ligand binding, part of the protein surface becomes buried and inaccessible, decreasing the SASA.

In this thesis, SASA is used as an MD feature due to its relationship with solvation entropy [4], [31], which, as described in Equation 2.6, forms part of the overall entropy driving PL binding. Upon binding, regions of the protein and ligand previously exposed to the solvent become buried, releasing solvent molecules. This release contributes favorably to the solvation entropy term, $\Delta S_{\text{solv}}$. Therefore, capturing this behavior through SASA measurements can provide important information for predicting binding affinity.

### 2.3.4 Hydrogen Bonds

A hydrogen bond is an attraction between or within the same molecule [32]. Hydrogen bonds occur when a hydrogen atom is connected to an atom like nitrogen, oxygen, or fluorine [33]. This pulls on the shared electrons, making the hydrogen slightly positively charged. When hydrogen is slightly positively charged, it attracts other nearby nitrogen, oxygen, or fluorine atoms. This interaction helps keep the molecules close together.

The reason for considering hydrogen bonds as a feature in MD simulations is that a higher number of hydrogen bonds between the ligand and the protein can indicate stronger binding and a more stable PL complex [18], [21]. Another motivation is the direct connection between hydrogen bonding and enthalpy [4]. As discussed in Section 2.1, enthalpy influences PL binding.

### 2.3.5 Interaction Energy

Van der Waals and electrostatic interactions are both types of non-covalent interactions that occur between the protein, ligand, and surrounding solvent. These interactions are enthalpic and contribute to the net enthalpy change ($\Delta H$), which in turn influences the Gibbs free energy (see Equation 2.6) and thus the binding affinity. The Coulombic interaction energy, representing the electrostatic interac-

tion energy, is the sum of all pairwise electrostatic interactions and is described by the last equation in Figure 2.3. The van der Waals interaction energy, calculated using the Lennard-Jones potential as described by the second-to-last equation in Figure 2.3, is implemented in GROMACS [34]. Van der Waals interactions are maximized when there is a tight fit between the ligand and the binding site, and even though weaker than electrostatic interactions, they are necessary for stabilizing high-affinity complexes [35], [36].

## 2.4 Machine Learning

This thesis will use supervised ML to predict binding affinity, which is the target variable. To perform this prediction, a dataset containing PL complexes is necessary. The goal is to develop an ML model that can take input features relevant to binding affinity (like the ones discussed in section 2.3) and produce a corresponding prediction [37]. This is similar to defining a mathematical function $f$ that represents the ML model, where the function takes features $x$ as input and outputs a predicted binding affinity, see figure 2.6.


**Figure 2.6:** ML model $f(x)$ that takes input features and outputs predicted binding affinity.

So, how is such a function determined? It is done through a process known as training, during which the function is adjusted to minimize the error in its predictions compared to the actual target values [37]. Once the training process is complete, an ML model is created that can predict binding affinity based on a given set of features. The next step is to evaluate the model's performance on PL complexes not included in the training set; this evaluation process determines how well the model generalizes to unseen examples.

### 2.4.1 Regression Models

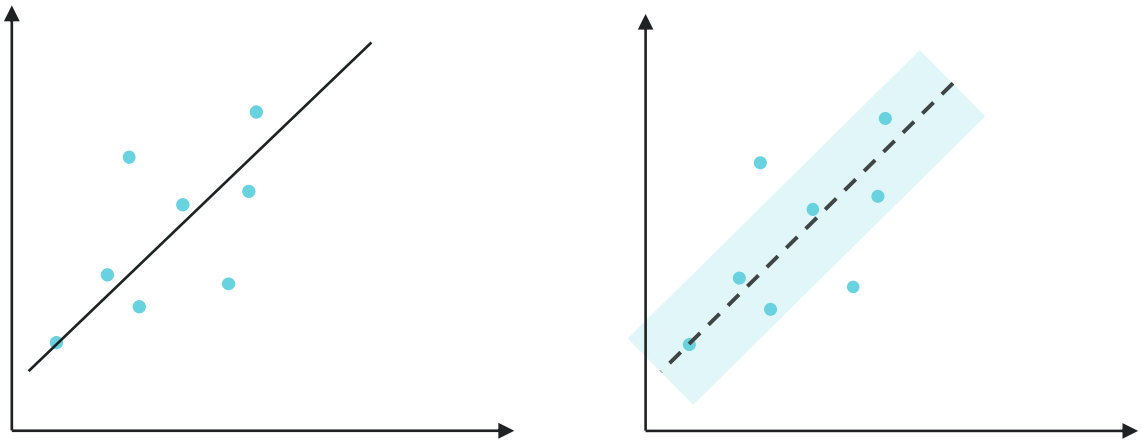
This section introduces the ML models used to predict binding affinity, a continuous numerical value. Since the task involves predicting numerical outputs, regression models are used [38].

Linear regression is a type of ML model that aims to find a relationship between the input features $\mathbf{x} = [x_1, x_2, ..., x_n]$ and a numerical target value, which in this thesis is the binding affinity [37]. The linear regression model is defined as:

$$y(\mathbf{x}) = \omega_0 + \omega_1 x_1 + \omega_2 x_2 + \cdots + \omega_n x_n \tag{2.8}$$

where $\omega_0, \omega_1, \omega_2, \cdots, \omega_n$ are the model parameters learned during training. The goal is to adjust the regression line, as shown in Figure 2.7a, so that the average distance from the predictions to the line is minimized (i.e., to minimize the error) [37].

Support Vector Machine (SVM) Regression is another ML model that will be used. As with Linear Regression, the goal is to find a relationship between the input features and the numerical target value. However, SVM Regression differs in its approach. Instead of minimizing the distance from all points (predictions) to the regression line, it focuses on reducing the distance from the predictions outside a specified margin (represented by a blue tube in Figure 2.7b) to the edge of this margin [37]. This approach makes SVM Regression less noise-sensitive, which helps the model's generalizability. Ridge Regression is another ML model used and builds



**(a)** Linear Regression **(b)** SVM Regression

**Figure 2.7:** Comparison of Linear Regression and SVM Regression approaches. (a) Linear Regression aims to minimize the distance from all points to the regression line. (b) SVM Regression focuses on minimizing the distance from points outside a specified margin to the edge of this margin.

upon Linear Regression by adding a regularization term. The regularization helps prevent overfitting to the training data (i.e., learning the noise), increasing the model's ability to generalize to unseen data [38]. This is achieved by controlling the model parameters and, consequently, the influence of each input variable, ensuring that no single variable dominates the predictions. Lasso Regression also builds on top of Linear Regression by using another type of regularization compared to Ridge Regression. It promotes sparsity in the model, encouraging only a few parameters to be non-zero. This effectively reduces the number of features the model uses for making predictions [38].

The next type of ML models that will be used are tree-based ones. Tree-based regression models make decisions based on specific rules [38]. The most basic is the Decision Tree Regressor, which constructs a decision tree as illustrated in Figure 2.8.

The model starts at the root node (green node) and traverses the tree by applying the rule at each node. It outputs the prediction once it reaches a leaf node (red node). Decision trees are capable of making non-linear predictions. However, on their own, decision trees are highly prone to overfitting.

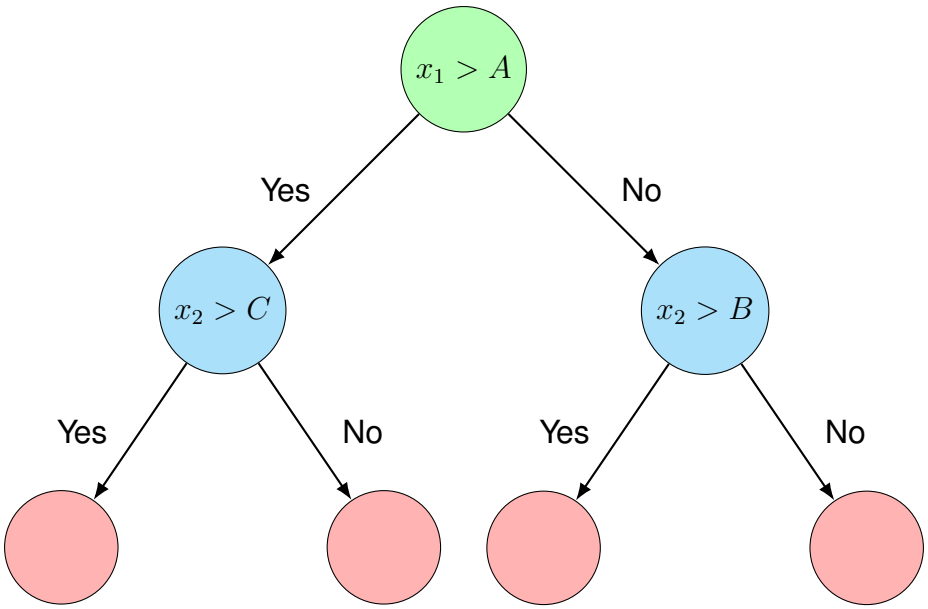



**Figure 2.8:** Decision Tree structure allows for decision-making based on specific rules, starting from the root node (green node) and continuing until reaching a leaf node (red node), where a prediction is made.

To address this, the Random Forest Regressor has been developed [38]. A random forest builds multiple decision trees, seen in figure 2.8, each trained on a different subset of the input features and data samples. This results in a collection of decision trees, and the final prediction is obtained by averaging the predictions from all trees.

Another powerful method is the Gradient Boosting, which builds trees sequentially rather than in parallel. Each new tree is trained to correct the errors made by the previous one. XGBoost [39], LGBM [40], and CatBoost [41] are powerful models that use gradient boosting in different innovative ways to make even better predictions.

### 2.4.2 Data Leakage

Data leakage is a common issue in ML applications [42], [43]. It occurs when information from the test set, data that should remain unseen during training, leaks into the training set. This allows the model to access information it would not have in a real-world application, often resulting in overestimating performance. Data leakage can occur during various stages, including data collection, pre-processing, and dataset splitting [43].

In this thesis, a potential source of data leakage is the presence of highly similar PL complexes in both the training and validation/test sets. This could cause the model

to effectively "see" test data during training, leading to overly optimistic results. The dataset is split randomly and based on the similarity between complexes to address this. See Section 3.1.4 for details on how the similarity-based split was performed. Comparing these two approaches highlights the importance of preventing data leakage.

### 2.4.3 Cross-Validation Techniques

Cross-validation is a commonly used technique for model selection in ML [44]. By evaluating on unseen data during training, a more robust ML model that is less likely to be overfitted can be created. One specific method is K-fold cross-validation, where the dataset is divided into K equally sized folds [37]. The model is trained on K-1 folds and evaluated on the remaining fold. This process is repeated K times, with each fold used once for validation. The final performance metric is then averaged across all K runs.

This thesis will use both K-fold cross-validation and Group K-fold cross-validation. Group K-fold takes into account predefined groups within the data. This is relevant when performing the similarity-based split described in Section 2.4.2, where similar complexes must not be split across training and validation sets.

Group K-fold ensures that all samples from the same group are kept together in either the training or validation set for each fold. The dataset is divided into G groups, which are then split into K folds. This approach helps prevent data leakage when related samples could otherwise appear in both sets.

### 2.4.4 Bayesian Hyperparameter Optimization

An ML model usually has some hyperparameters that must be set manually before training begins. These hyperparameters can significantly impact the model's performance and must be carefully tuned to achieve the best possible results.

Bayesian Optimization is an efficient method for hyperparameter tuning [45]. Suppose we have an ML model $f(x)$ with a set of hyperparameters to optimize. Bayesian Optimization begins by evaluating the model on a few selected combinations of hyperparameters [45]. Based on these evaluations, it constructs a probabilistic $f(x)$ model that maps hyperparameters to model performance. It then uses an acquisition function to select the next set of hyperparameters to evaluate. This function balances exploitation (choosing parameters with high expected performance) and exploration (trying parameters with more uncertainty). The more combinations that are evaluated, the better the model becomes at guiding the search for optimal hyperparameters.

### 2.4.5 Evaluation Metrics

Various evaluation metrics will be used to evaluate the performance of the different ML models described in Section 2.4.1. The first is Root Mean Squared Error (RMSE), which measures the average error size between the predicted and observed values. RMSE is on the same scale as the target value, and a lower RMSE indicates better performance, while a higher RMSE indicates worse performance. The RMSE is calculated as:

$$\text{RMSE} = \sqrt{\frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{n}} \tag{2.9}$$

where $y_i$ is the observed values and $\hat{y}_i$ is the predicted values.

The following evaluation metric is the Spearman rank correlation coefficient, $\rho$. This metric measures the strength and direction of the relationship between two ranked variables (predicted values and observed values). The ranking means that the highest value receives rank one, and lower values receive higher numerical ranks [46]. Spearman's $\rho$ is a nonparametric and distribution-free metric, meaning it does not assume any specific data distribution. It only requires a monotonic relationship (either increasing or decreasing). One calculates how much the ranks differ between the predicted ($\hat{Y}_i$) and observed ($Y_i$) sets, equation 2.10. $\rho$ takes a value between -1 and 1 where -1 indicates a perfect monotonic relationship and + 1 indicates a perfect negative one. 0 indicates no monotonic relationship.

$$\rho = 1 - \frac{6\sum_{i=1}^{n}(Y_i - \hat{Y}_i)^2}{n^2(n-1)} \tag{2.10}$$

The next evaluation metric is the Kendall rank correlation coefficient, $\tau$. This metric is also nonparametric and measures the association between two ranked variables [46]. The information provided by both Spearman and Kendall is quite similar. However, Spearman is more sensitive to errors because it takes the square of the differences (see Equation 2.10). It has been suggested that the lower Spearman and Kendall values should be considered [46]. Kendall's $\tau$ is calculated as:

$$\tau = \frac{n_c - n_d}{\frac{1}{2}N(N-1)} \tag{2.11}$$

Here, $n_c$ is the number of concordant pairs (i.e., for two data points $i$ and $j$, the pair is concordant if $predicted_i > predicted_j$ and $observed_i > observed_j$, or $predicted_i < predicted_j$ and $observed_i < observed_j$). $n_d$ is the number of discordant pairs (i.e., the pair is discordant if $predicted_i > predicted_j$ and $observed_i < observed_j$, or $predicted_i < predicted_j$ and $observed_i > observed_j$). $N$ is the total number of data points. This metric also takes values between -1 and 1, where -1 is a disagreement, 1 is an agreement, and 0 indicates no correlation.

The last metric used is the coefficient of determination $R^2$. This statistical measurement describes how much variation in the observed variables can be explained by the predicted values in a model [47]. A higher value of $R^2$ indicates a model with

better performance. The quantity can be calculated as follows

$$R^2 = 1 - \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{n}(y_i - \bar{y})^2} \tag{2.12}$$

where $\hat{y}_i$ is the predicted value, $y_i$ the observed value and $\bar{y}$ is the mean of the observed values.

### 2.4.6 Feature Reduction

Feature reduction is a pre-processing technique in ML. Feature reduction aims to reduce the dimensionality of the input data while preserving as much relevant information as possible [48]. Often, the feature space can be reduced without losing too much information, and in many cases, this can even increase the model's accuracy. The advantages of feature reduction are numerous, including removing redundant or noisy data, faster training times, and, in many cases, improved performance (i.e., increased predictive power).

# 3

# Methods

This section details the methodology used to address the central problem of developing a more accurate and faster approach for predicting binding affinity, as introduced in the preceding chapters. The project followed a sequential workflow, illustrated in Figure 3.1, progressing from the initial preparation of PL complex topologies (described in Section 3.1) to the execution of MD simulations (detailed in Section 3.2). Feature extraction is covered in Section 3.3. Following this, the development and evaluation of ML models are described in Section 3.4.

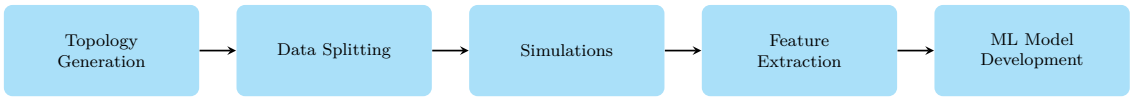



**Figure 3.1:** The workflow for predicting binding affinity encompasses PL complex preparation, MD simulations, feature extraction, and ML model development and evaluation.

## 3.1 Dataset and Data Processing

Two different datasets are used in this project. Section 3.1.1 details the initial dataset used for training and validating the ML models. Section 3.1.2 describes the test set used for the final evaluation of the ML models.

### 3.1.1 Initial Dataset Description

The initial dataset used in this study was developed in-house at AstraZeneca and consists of approximately 5,000 three-dimensional (3D) PL complexes, each based on X-ray crystallography. For each complex, the protein structure was provided in the PDB (Protein Data Bank) format, and the ligand was provided in the SDF (Spatial Data File) format, along with experimentally determined binding affinities expressed in $pK_b$ as described in Section 2.1.1 and docking scores. A specific preprocessing workflow was implemented to prepare each complex for MD simulations using GROMACS. First, the Ligands were protonated using RDkit, and then

topologies were generated using AmberTools (Antechamber) with a compatible force field and converted to the GROMACS format via ACPYPE. This step addressed the common issue of GROMACS force fields not recognizing specific ligands [26]. Protein topologies were prepared directly using GROMACS's pdb2gmx tool. The ligand and protein coordinates were combined into a single GROMACS input file, with the ligand information integrated into the system topology.

### 3.1.2 Test Set Description

The test set is similar to the initial dataset described in Section 3.1.1, with a few key differences. It consists of approximately 750 3D PL complexes, where the starting structures were generated through molecular docking aligned to an experimentally known reference ligand rather than being obtained directly from X-ray crystallography, as in the initial dataset. Another notable difference is that, in the test set, each protein is paired with a congeneric series of ligands, meaning the scaffold remains unchanged. In contrast, specific parts of the ligand are modified, reflecting the lead optimization phase of drug development. In contrast, the initial dataset consisted of a more diverse set of ligands. Each test set complex includes FEP-calculated and experimentally measured binding affinities. The FEP-calculated binding affinities were performed using Schrödinger's FEP+ [49] but will only be referred to in this thesis as FEP. The dataset was prepared using the same protocol as the initial dataset and serves as a final benchmark to evaluate the ML model developed in this thesis against a more accurate computational method: FEP.

### 3.1.3 Data Splitting

After generating the PL complexes described in 3.1.1, the complexes were split into training and validation sets using two strategies. The first approach was a standard random split, allocating 80% of the complexes to the training set and 20% to the validation set. The second approach applied a similarity-based splitting, which is detailed below. The different split is due to the effects of data leakage, an issue discussed in section 2.4.2 and how it can be prevented when training ML models.

### 3.1.4 Similarity-Based Dataset Splitting

A clustering approach was used to group PL complexes based on calculated similarity between complexes. These clusters were then used for the similarity-based split. The steps are as follows:

### Step 1: Similarity Calculation

**Ligand Similarity:** The RDKit library was used to compute the Tanimoto similarity between ligand molecules. Resulting in an $n \times n$ similarity matrix, where each element represents the Tanimoto coefficient between ligand pairs. The Tanimoto coefficient is one of the most widely used metrics for assessing chemical similarity [50], and is used in this thesis to quantify the similarity between ligands.

**Protein Similarity:** Using the Biopython package, global protein sequence alignments were performed to create an $n \times n$ similarity matrix with the alignment scores for each protein pair. This approach was used by Ragoza et al. [51].

### Step 2: Distance Matrix Construction

These similarity scores were transformed into distance measures:

- **Ligand Distance Matrix:** For each ligand pair, the distance was calculated by subtracting the Tanimoto similarity score from 1.
- **Protein distance matrix:** Similarly, for each protein pair, the distance was derived by subtracting the alignment score from 1.

The ligand and protein distance matrices were stacked using NumPy, forming a 3D array for combined distance evaluation.

### Step 3: Threshold Application

A boolean similarity matrix was generated to indicate whether pairs of PL complexes are considered similar. Two complexes are defined as similar if they satisfy either of the following conditions:

1. The distance between their protein structures is less than 0.5, **or**
2. The distance between their ligands is greater than 0.9 **and** the distance between their protein structures is less than 0.7.

These distance thresholds were originally introduced by Ragoza et al. [51] for clustering PL complexes and are adopted without modification in this work.

### Step 4: Clustering with GROMOS

The GROMOS clustering algorithm, as described by Daura et al. [52] (initially developed for clustering MD trajectories), was used to process the boolean matrix, grouping ligand-protein complexes into clusters.

**Step 5: Dataset split**

The clustered complexes were separated into training (approximately 80%) and validation (approximately 20%) sets. Clusters were kept intact, meaning that only whole clusters, rather than individual complexes, were divided between the different sets. This approach was used to avoid data leakage.

## 3.2 Molecular Dynamics Simulations

This section outlines the setup and execution of the MD simulations, which is the third step in the methodology. It describes the simulation configuration, the parameters used, the energy minimization, equilibration procedures, and the production run.

### 3.2.1 Simulation Setup

MD simulations were performed using GROMACS 2024, which was selected for its design for protein systems, comprehensive documentation, and broad use within the scientific community. These simulations were carried out on AstraZeneca's high-performance computing cluster, leveraging GROMACS's efficient support for graphical processing units (GPUs) and parallel processing to handle the dataset of 5,000 PL complexes.

The simulation protocol followed the standard workflow outlined in Justin Lemkul's MD tutorial [26], with minor adjustments. This approach was chosen because the project aimed to conduct short simulations for high-throughput analysis, thus eliminating the need for a specialized setup. The AMBER99SB-ILDN force field was applied to the proteins, while General AMBER ForceField (GAFF2) was used for the ligands. Each PL complex was solvated in a dodecahedral box using the TIP3P water model, ensuring a minimum distance of 1.5 nm between the solute and the box boundaries. Solvation was performed using the gmx solvate tool. To achieve charge neutrality, $Na^+$ and $Cl-$ ions were added by replacing water molecules using the gmx genion tool, and the system topology was updated accordingly. The resulting solvated and neutralized system served as the initial configuration for energy minimization.

### 3.2.2 Simulation Parameters

The leap-frog algorithm was employed to integrate the equations of motion with a time step of 2 fs. The time step of 2 fs worked because the bonds between atoms that involved hydrogens were constrained using the LINCS algorithm. Long-range electrostatic interactions were treated using the Particle Mesh Ewald (PME) method.

For short-range van der Waals interactions, a cutoff distance of 1.2 nanometers (nm) was applied using the Verlet cutoff scheme. Throughout all simulation stages, positional restraints were applied to the protein backbone atoms using a force constant of 200 kJ $mol^{-1}$ $nm^{-2}$ in each Cartesian direction. This was done to ensure the protein remained close to the initial X-ray structure during the short simulations. Initial atomic velocities were assigned based on a Maxwell-Boltzmann distribution corresponding to the target simulation temperature.

### 3.2.3 Energy Minimization and Equilibration

**Energy Minimization** was performed using the steepest descent algorithm until the maximum force on any atom reached 10 kJ $mol^{-1}$ $nm^{-1}$. A step size of 0.01 nm was used, with a maximum of 50,000 minimization steps.

**NVT Equilibration with Simulated Annealing:** The systems were equilibrated under a constant volume (NVT) ensemble for 100 ps with a time step of 2 fs. Simulated annealing gradually heated the system from 0 K to 300 K throughout 10 ps; this one was done to keep the simulations close to the x-ray structure. The temperature was controlled using the velocity-rescaling (V-rescale) thermostat, with separate coupling groups defined for the PL complex and the solvent (water and ions).

**NPT Equilibration:** Following NVT equilibration, the system underwent further equilibration under constant pressure (NPT) conditions for 100 ps to allow volume adjustments and achieve the appropriate system density. The pressure was maintained at 1 bar using the C-rescale barostat with isotropic box scaling. Temperature coupling was maintained using the same thermostat settings as in the NVT equilibration phase. The pressure coupling time constant was set to 2 ps, and reference coordinate scaling was applied.

### 3.2.4 Production and Energy Calculation Runs

The production simulations employed an integration time step of 2 fs, consistent with the parameters used in Lemkul's tutorial [26]. The simulation duration was set to 2 nanoseconds (ns), and three independent replicas were run for each system to enhance the reliability of the results [53].

A rerun of the production simulations was necessary to calculate interaction energies. This was primarily due to the lack of GPU support for calculating interaction energies simultaneously with the production run in GROMACS. As recommended by the GROMACS software documentation, a separate rerun was performed, during which specific atom groups were defined, for which the interaction energies were to be calculated. More information about the extraction of interaction energies in Section 3.3.5

Once interaction energies were calculated, the solvent molecules were removed from the trajectory to reduce the size of the simulation files.

## 3.3 Feature extraction

This section outlines how the MD features introduced in Section 2.3 were extracted from the MD trajectories generated in the simulations described in Section 3.2. These features were then transformed into a format suitable for comparison across different PL complexes. As discussed in Section 2.1, the extracted features are related to entropy and enthalpy and include the following: RMSF, dihedral angles, SASA, hydrogen bonds, interaction energy, ligand properties, and docking score.

Different software packages were used to extract the various features from the simulation trajectories. The software and packages included, MDAnalysis [54], GROMACS [55], GNINA [15] and RDKit [56]. An overview of each feature and the corresponding software or package used for its extraction is provided in Table 3.1.

**Table 3.1:** Overview of extracted features and the corresponding software or packages used.

| **Feature** | **Software/Package** |
|---|---|
| RMSF | MDAnalysis |
| Dihedral Angles | MDAnalysis, GROMACS |
| SASA | GROMACS |
| Hydrogen Bonds | MDAnalysis |
| Interaction Energy | GROMACS |
| Ligand Properties | RDKit |
| Docking Score | GNINA |

### 3.3.1 Root Mean Square Fluctuations

RMSF, a measure of atomic flexibility described in Section 2.3.1, was analyzed near the ligand to capture the dynamic behavior of the binding site. Two shells around the ligand were considered: protein atoms within 0–5 Å and those within 5–10 Å, as illustrated in Figure 3.2.

The RMSF values of the 20 closest protein atoms to the ligand were extracted for each shell. If a shell contained fewer than 20 atoms, the resulting RMSF vector was zero-padded to maintain consistent dimensionality. Three distinct feature representations were derived from this 20-element RMSF vector for each shell:

1. A scalar feature calculated as the sum over all 20 atoms in the vector of the natural logarithm of the product of the RMSF value ($rmsf_i$) and the square root of the atom's mass ($\sqrt{m_i}$), i.e., $\sum_{i=1}^{20} \ln(rmsf_i \sqrt{m_i})$.

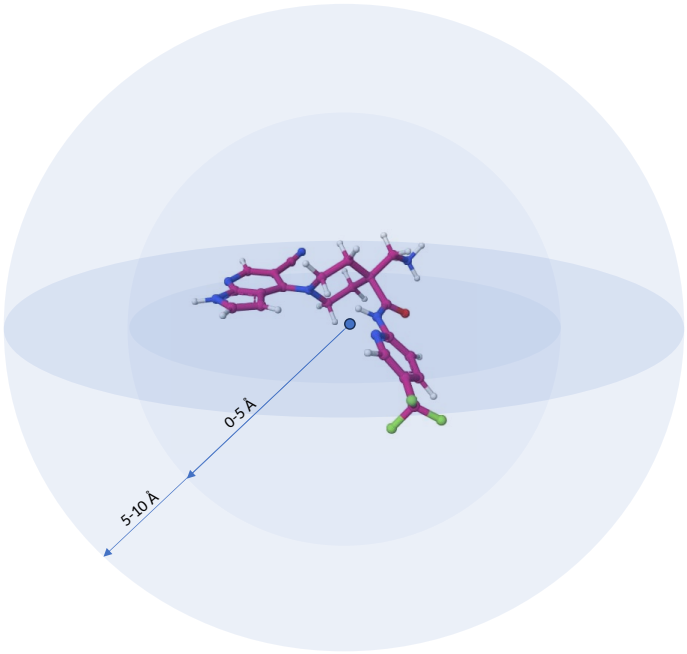



**Figure 3.2:** Visualization of two shells around the ligand highlighting the different atom ranges analyzed for RMSF feature extraction.

2. The entire 20-element RMSF vector was used directly as a feature vector.
3. The fluctuations of the 20 atoms within the vector were discretized into five bins with the following boundaries (in Å): 0–0.5, 0.5–1, 1–1.5, 1.5–2, 2–2.5, and $> 2.5$. The resulting feature was a 6-element vector representing the count of atoms falling into each bin.

The scalar feature, as described for the protein in step one, was also calculated for the ligand.

This procedure yielded 27 features per shell (2 scalars + 20 vector elements + 6 bin counts), resulting in 54 RMSF-related features for each PL complex.

### 3.3.2 Dihedral Angles

The extraction of dihedral angle features, as described in Section 2.3.2, followed a technique similar to that used for the RMSF analysis. Two shells surrounding the ligand, defined by distances of 0–5 Å and 5–10 Å, were considered, as illustrated in Figure 3.2. For each shell, all dihedral angles with at least one atom located within the specified distance were identified. If any atom defining the dihedral angle resided within a shell, the values of all four atoms involved in that dihedral were extracted.

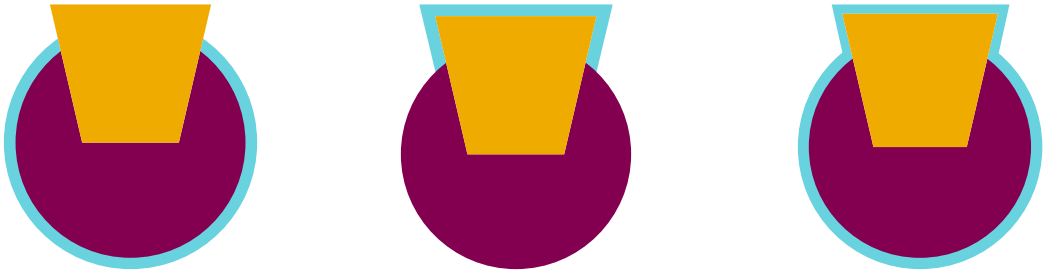
Next, the fluctuation of each identified dihedral angle was computed due to its relevance for conformational entropy [28]. These fluctuations were then discretized into six bins based on their magnitude (in degrees): 0–10, 10–20, 20–30, 30–40, and

greater than 40. This discretization helps categorize the dihedral motion and identify conformational flexibility. A 6-element vector was generated for each shell, representing the count of dihedral angles falling into each bin. Additionally, the average fluctuation value of all dihedral angles identified within each shell was calculated, resulting in a single scalar feature per shell.

This procedure yielded seven features per shell (6 bin counts + 1 average fluctuation), totaling 14 dihedral angle-related features for each PL complex.

### 3.3.3 Solvent Accessible Surface Area

This feature captures the SASA, which quantifies the solvent-exposed surface of molecular components, as described in Section 2.3.3. Figure 3.3 illustrates the PL complex and the different methods of SASA extraction, with the blue shading indicating the areas where SASA measurements were taken. The following SASA measurements were calculated: total protein SASA, as shown in Figure 3.3a, along with individual contributions from hydrophobic and hydrophilic regions. Figure 3.3b shows the total ligand SASA, while Figure 3.3c presents the combined SASA of the PL complex. Additionally, the fluctuation of each of these five SASA values over the simulation trajectory was computed, resulting in 10 SASA-related features comprising two features per category.



**(a)** Total protein SASA **(b)** Ligand SASA **(c)** PL SASA

**Figure 3.3:** Visual representation of SASA calculations for (a) Protein, (b) Ligand, and (c) Combined complex demonstrating their solvent accessible areas.

### 3.3.4 Hydrogen Bonds

The hydrogen bond feature, as described in Section 2.3.4, was extracted by quantifying the number of hydrogen bonds formed between the protein and the ligand at each frame of the MD trajectory. Four metrics were calculated:

1. The fluctuation of the number of hydrogen bonds over the simulation.
2. The average number of hydrogen bonds per frame.

3. The total number of hydrogen bonds observed throughout the simulation.
4. The maximum number of hydrogen bonds observed in any single frame.

### 3.3.5 Interaction Energies

Interaction energies between PL and ligand-solvent were extracted from the MD trajectories generated during the production rerun (as detailed in Section 3.2). These energies were calculated using the `gmx energy` module within the GROMACS software, with a focus on the non-bonded van der Waals (Vdw) and Coulombic (Coul) components.

For both the PL and ligand-water pairs, the following analyses were performed:

1. **Total Interaction Energy:** The sum of the average van der Waals and average Coulombic interaction energies ($\langle Vdw \rangle + \langle Coul \rangle$) over the simulation.
2. **Fluctuation of Van der Waals Energy:** The standard deviation of the van der Waals interaction energy ($\sigma_{Vdw}$) over the simulation.
3. **Fluctuation of Coulombic Energy:** The standard deviation of the Coulombic interaction energy ($\sigma_{Coul}$) over the simulation.

This approach yielded six interaction energy-based features: three for the PL interaction and three for the ligand-water interaction.

### 3.3.6 Docking Score

The Molecular Docking Software, described in Section 2.2.1, provides several outputs that will be used as features for the ML model:

- **minimizedAffinity** – The regular docking score for the ligand after re-docking.
- **minimizedRMSD** - Root Mean Square Deviation of the re-docked ligand compared to the original x-ray ligand pose.
- **CNNscore** – The probability that the re-docked pose is closer than 2Å to the in x-ray.
- **CNNaffinity** – The binding affinity estimated by GNINA.
- **CNN_VS** – CNNscore × CNNaffinity.
- **CNNaffinity_variance** – The uncertainty associated with the affinity estimation.

### 3.3.7 Ligand Properties

Eighty-three ligand properties were derived using the rdkit.Chem.rdMolDescriptors module and included as additional features alongside MD-derived features and docking scores. Boyle et al. [57] demonstrated that ML could enhance binding affinity prediction by combining molecular docking information with ligand descriptors. Their results showed improved predictive performance when such descriptors were incorporated. Based on these findings, including ligand descriptors in the feature set is a natural choice. The specific descriptors used are listed in the appendix.

### 3.3.8 Feature Reduction

Correlation-based feature selection [48] was used to reduce the number of features. The goal was to eliminate redundant features, those highly correlated with other features, and those that do not correlate with the observed value. This was assessed using the Spearman rank correlation coefficient, $\rho$, as described in Section 2.4.5. All features with an absolute correlation coefficient greater than 0.1 with the observed value were kept, while features with an absolute correlation greater than 0.95 with another feature were removed. This method was applied to all features mentioned in Sections 3.3.1 - 3.3.7, resulting in a total of 87 features.

### 3.3.9 Summary of Features

A summary of the number of features extracted and their respective types is presented in Table 3.2. Note that after *Total features* is also how many features were left after the feature reduction introduced in Section 3.3.8.

**Table 3.2:** Summary of features.

| **Feature Category** | **Number of Features** |
|---|---|
| RMSF Related Features | 54 |
| Dihedral Angle Related Features | 14 |
| SASA Related Features | 10 |
| Hydrogen Bond Related Features | 4 |
| Interaction Energy Related Features | 6 |
| Docking Features | 6 |
| Ligand Descriptor Features | 83 |
| **Total Features** | **183** |
| **Correlation-Based Feature Set** | **87** |

## 3.4 Machine Learning Model Development

This section outlines the steps to select an ML model, which uses the features described in Section 3.3 to predict binding affinity. As Figure 3.4 illustrates, the ML model development process consists of five blocks.

Model evaluation will be conducted using the random train-validation split and similarity split methods described in Section 3.1. For models trained with the random train-validation split, a 5-fold cross-validation approach will be used. In contrast, a Group 5-fold cross-validation will be applied when using the similarity split to prevent data leakage during the cross-validation process.


**Figure 3.4:** Overview of the ML workflow used to develop an ML model for predicting binding affinity.

### 3.4.1 Baseline Model Description

Two baseline models were constructed to evaluate the contribution of the MD-derived features. Considering the thesis's objective of enhancing binding affinity prediction beyond molecular docking, a logical baseline used a simple linear regression model with only the CNNaffinity score (the predicted binding affinity from GNINA) as the first baseline model. The second baseline model uses a simple linear regression with the minimizedAffinity (predicted binding affinity before being re-evaluated using GNINA) as the input feature. These two baselines will be evaluated for both the random and similarity-based split. The performance of these baselines on the validation set will indicate whether incorporating MD-derived features and ligand properties improves predictive capability and whether more complex models lead to better performance.

### 3.4.2 Evaluation of Candidate Models

Regression models were evaluated using their default settings to identify a suitable model architecture for predicting binding affinity. The candidate models included:

These models were trained using the complete feature set described in Section 3.3 and evaluated through cross-validation on the training set, both for the Random split and the Similarity-based split. Each model's performance was reported as the average across all folds, using the following metrics: R-squared ($R^2$), RMSE, Kendall's $\tau$, and Spearman's $\rho$, as these are commonly used for evaluating binding affinity predictions [3].

- CATBoostRegressor
- XGBRegressor
- LGBMRegressor
- Random Forest Regressor
- Decision Tree Regressor
- Linear Regression
- Ridge Regression
- Lasso Regression
- SVM Regressor

During cross-validation, the model that showed the best overall performance across these metrics (i.e., high $R^2$, low RMSE, high $\tau$, and high $\rho$) was selected for further optimization. This selected model underwent hyperparameter tuning using Bayesian Optimization, as described in Section 2.4.4. The reason for using Bayesian Optimization for hyperparameter tuning and not Grid Search and Random Search is because those methods do not consider previous results, which Bayesian Optimization does [58]. This results in faster convergence for Bayesian Optimization, as it uses previously learned information.

### 3.4.3 Feature Set Evaluation

Following the selection of the best ML model (as described in Section 3.4.2), the predictive power of different feature sets was evaluated to understand the contribution of various features. Beyond using all features (as initially done during candidate model evaluation in Section 3.4.2), the following subsets were examined:

1. **Docking Features**: These features, detailed in section 3.3.6, represent information derived solely from molecular docking.
2. **MD Features**: This set includes only the features extracted from the MD trajectories, as described in sections 3.3.1-3.3.5.
3. **Docking Features + MD features**: This combined set incorporates the features from both feature sets 1 and 2.
4. **Ligand Properties + MD features**: This set combines the MD features (set 2) with the ligand properties described in section 3.3.7.
5. **Correlation-Based Features**: This feature set is described in Section 3.3.8 and is based on correlation analysis of all features.

These feature sets were used to investigate how adding MD-derived features to the docking features impacts the binding affinity prediction. This involved assessing the individual performance of docking and MD features, followed by observing the effect of their gradual combination (both with and without ligand properties), to quantify the value each feature set contributes to the model's predictive capabilities.

The ML model was trained on each feature set using hyperparameters found through Bayesian Optimization and then evaluated using the validation set for each feature set. From now on, only the similarity-based split will be used. This is because it represents more of the model's true performance.

### 3.4.4 Analysis of Selected Model

Following the model selection in section 3.4.2 and the feature set selection in section 3.4.3, an analysis of the features most influential in the model's predictions was done. This analysis aimed to identify which types of features contribute most to binding affinity prediction. Additionally, a final comparison was performed between the selected model and the affinity predictions from docking (minimizedAffinity) and improved prediction (GNINA), using various evaluation metrics. This evaluation offers insights into the relative performance of the developed model compared to traditional molecular docking methods.

## 3.5 Final Evaluation

For the final evaluation, the best-performing ML model identified in section 3.4.2 was trained using the optimal feature set determined in section 3.4.3. The model's performance was then assessed using the independent test set described in section 3.1.2. The evaluation strategy was twofold. First, a comprehensive assessment of the model's performance on the entire test set was conducted, comparing it with affinity predictions from docking (minimizedAffinity), re-docked prediction (GNINA), and FEP using the different evaluation metrics. Second, protein-specific predictions were performed to investigate the model's performance across different proteins. For each protein, binding affinity predictions from the ML model were compared with those from docking, GNINA, and FEP. This evaluation provided insights into how the model's performance varied across different protein targets.

# 4

# Results

## 4.1 Performance of Baseline Models

The results from the baseline models, shown in Table 4.1, indicate that CNNaffinity consistently outperforms minimizedAffinity across all evaluation metrics for both splitting strategies. Interestingly, when using minimizedAffinity, the similarity-based split yielded slightly better performance than the random split.

**Table 4.1:** Performance comparison of baseline models using different splitting strategies.

| **Split** | **Feature** | **$R^2$** | **RMSE** | **Kendall's $\tau$** | **Spearman's $\rho$** |
|---|---|---|---|---|---|
| Random | CNNaffinity | 0.45 | 1.28 | 0.45 | 0.63 |
| Random | minimizedAffinity | 0.29 | 1.45 | 0.36 | 0.52 |
| Similarity | CNNaffinity | 0.39 | 1.31 | 0.41 | 0.58 |
| Similarity | minimizedAffinity | 0.30 | 1.39 | 0.36 | 0.53 |

## 4.2 Evaluation of Candidate Models

Different regression models were evaluated on both the Random split (Table 4.2) and the similarity-based split (Table 4.3).

On the Random split (Table 4.2), the tree-based models; CatBoost Regressor, LGBM Regressor, XGB Regressor, and Random Forest, generally demonstrated strong performance across all metrics. They showed higher $R^2$ values (ranging from 0.639 to 0.658), lower RMSE values (ranging from 0.990 to 1.036 $pK_b$), and higher ranking correlation coefficients ($\tau$ ranging from 0.585 to 0.597, and $\rho$ ranging from 0.772 to 0.785) compared to the other models. SVM Regressor also showed competitive results. In contrast, Linear Regression and Ridge Regression displayed moderate performance, while Decision Tree and Lasso Regression consistently produced the poorest results across all evaluation metrics.

A similar trend was observed for the similarity-based split (Table 4.3). The tree-based models (CatBoostRegressor, Random Forest, LGBMRegressor, and XGBRe-

**Table 4.2:** Candidate model evaluation results (random split). The row highlighted in gray corresponds to the best-performing model, CatBoost Regressor.

| **Model** | $R^2$ | RMSE | $\tau$ | $\rho$ |
|---|---|---|---|---|
| CatBoost Regressor | 0.658 | 0.990 | 0.597 | 0.785 |
| XGB Regressor | 0.625 | 1.036 | 0.569 | 0.757 |
| LGBM Regressor | 0.648 | 1.005 | 0.587 | 0.776 |
| Random Forest | 0.639 | 1.018 | 0.585 | 0.772 |
| Decision Tree | 0.237 | 1.479 | 0.424 | 0.588 |
| Linear Regression | 0.554 | 1.131 | 0.521 | 0.705 |
| Ridge Regression | 0.552 | 1.134 | 0.519 | 0.703 |
| Lasso Regression | 0.107 | 1.601 | 0.450 | 0.625 |
| SVM Regressor | 0.619 | 1.046 | 0.576 | 0.764 |

gressor) again showed good performance across the metrics, with relatively higher $R^2$ values (ranging from 0.535 to 0.554), lower RMSE values (ranging from 1.104 to 1.218 $pK_b$), and better ranking correlation ($\tau$ ranging from 0.507 to 0.533, and $\rho$ ranging from 0.690 to 0.716) compared to the linear models and the Decision Tree and Lasso Regressors, which struggled across all metrics. SVM Regressor's performance was also within the range of the better-performing models.

**Table 4.3:** Candidate model evaluation results (similarity-based split). The row highlighted in gray corresponds to the best-performing model, CatBoost Regressor.

| **Model** | $R^2$ | RMSE | $\tau$ | $\rho$ |
|---|---|---|---|---|
| CatBoost Regressor | 0.554 | 1.104 | 0.533 | 0.716 |
| XGB Regressor | 0.484 | 1.218 | 0.481 | 0.658 |
| LGBM Regressor | 0.536 | 1.164 | 0.507 | 0.690 |
| Random Forest | 0.535 | 1.148 | 0.510 | 0.691 |
| Decision Tree | 0.094 | 1.601 | 0.376 | 0.532 |
| Linear Regression | 0.504 | 1.187 | 0.497 | 0.680 |
| Ridge Regression | 0.448 | 1.228 | 0.467 | 0.644 |
| Lasso Regression | 0.050 | 1.643 | 0.458 | 0.628 |
| SVM Regressor | 0.524 | 1.156 | 0.512 | 0.695 |

All models except Decision Tree and Lasso Regression outperformed the baseline models (Table 4.1). Considering the performance across the various evaluation metrics on random and similarity-based splits, CatBoost Regressor was selected as the model for further hyperparameter optimization. It is also worth noting that performance dropped when using a similarity-based split compared to a random split, as expected.

## 4.3 Optimization of Selected Model

As the CatBoost Regressor demonstrated the best overall performance in the candidate model evaluation, hyperparameter tuning was performed to increase its performance. This optimization used Bayesian Optimization with 100 iterations for both the Random and Similarity-based splits. The best hyperparameter values identified for each split are presented in Table 4.4. The meaning of each hyperparameter, as provided in the CatBoost documentation, is described below:

- **Learning Rate:** How fast the model learns.
- **Iterations:** Number of trees that can be built.
- **Depth:** How deep the trees can grow.
- **L2 Leaf Reg:** L2-regularization (prevent overfitting).
- **Min Data in Leaf:** The minimum number of training samples in a leaf.
- **Random Strength :** Used to prevent overfitting the model.
- **Border Count:** Number of trees that can be built.

**Table 4.4:** Hyperparameters identified through hyperparameter tuning for the Random Split and Similarity-Based Split.

| **Hyperparameter** | **Random Split** | **Similarity-Based Split** |
|---|---|---|
| Learning Rate | 0.030 | 0.027 |
| Iterations | 1000 | 739 |
| Depth | 10 | 10 |
| Min Data in Leaf | 1 | 30 |
| Border Count | 32 | 32 |
| L2 Leaf Reg | 0.001 | 0.001 |
| Random Strength | 0.001 | 0.001 |

The performance results before and after hyperparameter tuning of the CatBoost regressor for both the random split and the similarity-based split are shown in Figure 4.1 and Figure 4.2, respectively. In both figures, we observe a slight improvement in performance for $R^2$, Kendall's $\tau$, and RMSE, while Spearman's $\rho$ remains unchanged. As expected, there is a consistent decrease in performance when transitioning from the random split to the similarity-based split.

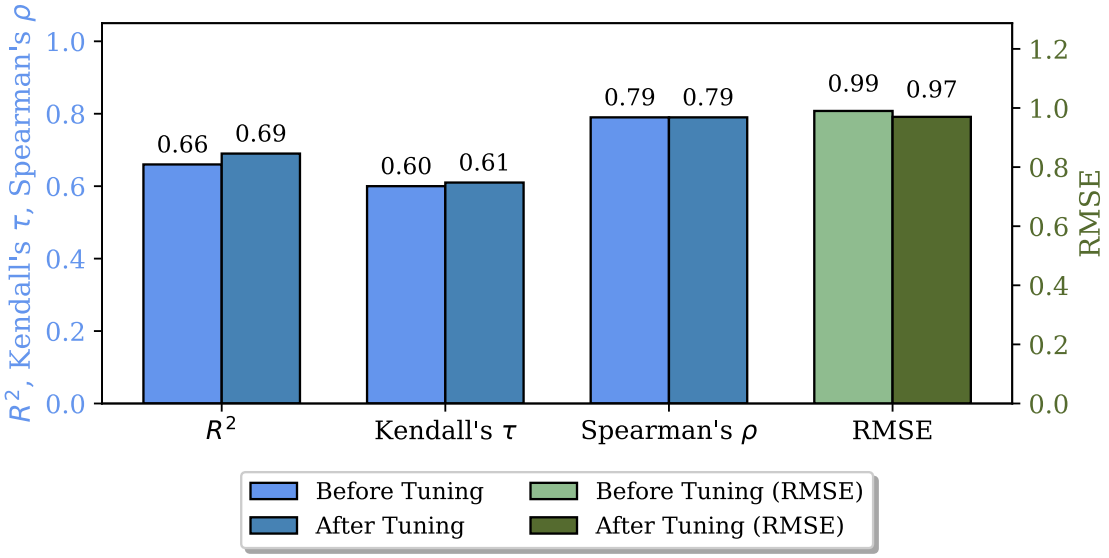



**Figure 4.1:** Model performance before and after hyperparameter tuning across multiple metrics using the random split. For RMSE, lower values indicate better performance, whereas for $R^2$, Kendal's $\tau$, and Spearman's $\rho$, higher values indicate better performance.

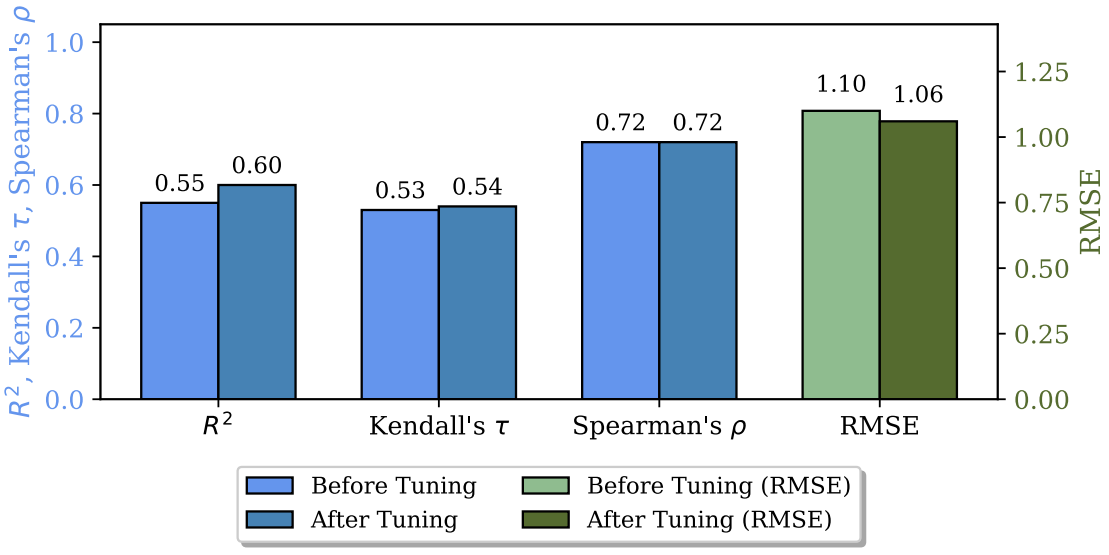



**Figure 4.2:** Model performance before and after hyperparameter tuning across multiple metrics using the similarity-based split. For RMSE, lower values indicate better performance, whereas for $R^2$, Kendall's $\tau$, and Spearman's $\rho$, higher values indicate better performance.

## 4.4 Feature Set Evaluation

An evaluation of the different feature sets described in Section 3.4.3 was conducted, with the results summarized in Table 4.5. Using all available features delivered strong performance across all metrics, outperforming the individual use of Docking and MD features. Notably, MD features performed better than Docking features.

Combining Docking and MD features further improved performance compared to using them separately, but combining Ligand Properties with MD features achieved an equally strong, if not slightly better, performance. This combination matched the previous one in terms of RMSE and exceeded it in ranking metrics ($\tau$ and $\rho$) despite excluding docking scores.

Applying correlation-based feature reduction resulted in a model that performed slightly better than the complete feature set, while using fewer features. It achieved the same $R^2$ (0.60), a slightly lower RMSE (1.05 vs 1.06), and subtle gains in $\rho$.

Based on these findings, the correlation-based feature set will be used in the final model due to its comparable (and slightly improved) performance with reduced complexity.

**Table 4.5:** Performance comparison of different feature sets based on $R^2$, RMSE, Kendall's $\tau$, and Spearman's $\rho$. The best-performing feature set was the correlation-based features, highlighted in gray.

| **Feature Set** | $R^2$ | RMSE | $\tau$ | $\rho$ |
|---|---|---|---|---|
| Docking Features | 0.44 | 1.25 | 0.43 | 0.61 |
| MD Features | 0.51 | 1.17 | 0.46 | 0.63 |
| Docking Features + MD Features | 0.58 | 1.09 | 0.51 | 0.69 |
| Ligand Properties + MD Features | 0.57 | 1.09 | 0.52 | 0.71 |
| Correlation-Based Features | 0.60 | 1.05 | 0.54 | 0.73 |
| All Features | 0.60 | 1.06 | 0.54 | 0.72 |

## 4.5 Analysis of Selected Model

As the CatBoost Regressor was the the best-performing model, and the most effective feature set was the correlation-reduced feature set, a more thorough analysis of the model was conducted. First, CatBoost's built-in feature importance method was used, which outputs a list of features along with their corresponding importance scores. These scores indicate how frequently a feature was used to make better predictions. The importance values for all features sum up to 100.

In Table 4.6, the feature importances are grouped according to the feature types introduced in Section 2.3. The MD features received a total importance score of 43.5, the Docking features 6.97, and the Ligand Properties 49.77. Among the MD features, Dihedral Angles appeared to be the most influential, followed by RMSF and SASA. For the Docking features, the most important ones were minimizedRMSD and CNN_variance. Another thing to keep in mind is that, as shown in Table 4.6, all three feature categories are part of the correlation-based features introduced in Section 3.3.8.

**Table 4.6:** Feature importance grouped by MD features, Docking features, and Ligand properties. The total of MD Features, Docking Features, and Ligand Properties sums up to $\approx 100$.

| **Feature Group** | **Importance** |
|---|---|
| **MD Features** | |
| Dihedral Angles | 27.28 |
| RMSF | 7.42 |
| Hbonds | 1.90 |
| SASA | 5.25 |
| Interaction Energy | 1.60 |
| **Total MD** | **43.50** |
| **Docking Features** | |
| minimizedRMSD | 2.04 |
| minimizedAffinity | 0.61 |
| CNNscore | 0.59 |
| CNNaffinity | 0.91 |
| CNN_VS | 0.37 |
| CNN_variance | 2.44 |
| **Total Docking** | **6.97** |
| **Ligand Properties** | |
| **Total Ligand** | **49.77** |

The performance of the CatBoost model was also evaluated by comparing it against Docking and GNINA binding affinity predictions using the validation set. As shown in Figure 4.3, the CatBoost Regressor model outperformed both traditional Docking and GNINA predictions across all metrics. Note that the right y-axis is used for RMSE, where a lower value indicates better performance, in contrast to the metrics $R^2$, Kendall's $\tau$, and Spearman's $\rho$.

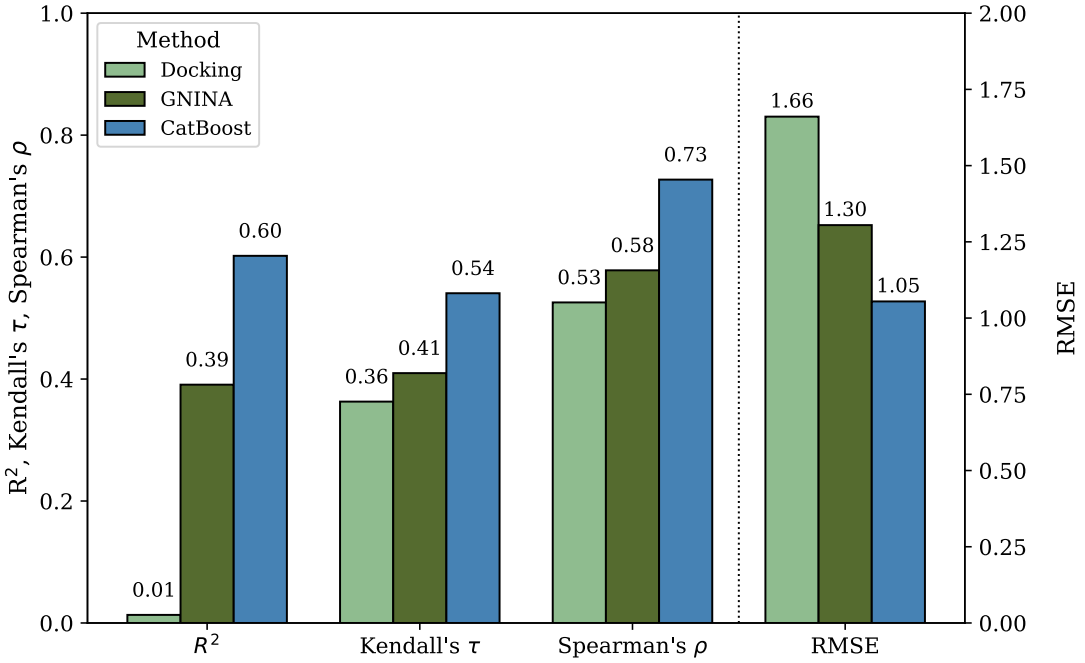



**Figure 4.3:** Comparison of model performance, showing the predicted values from the CatBoost model evaluated on the validation set, alongside the affinity predictions from traditional Docking and GNINA. The left y-axis represents $R^2$, Kendall's $\tau$, and Spearman's $\rho$, while the right y-axis corresponds to RMSE.

## 4.6 Final Evaluation

The final evaluation of the CatBoost model is presented in Figure 4.4. Consistent with the results from the validation set (Figure 4.3), the CatBoost model outperforms Docking and GNINA across all evaluation metrics. Additionally, the test set includes binding affinity predictions obtained using FEP, represented by the orange bar. As shown, FEP achieves the highest performance overall, outperforming all other methods, including the CatBoost model, as expected. Note that the right y-axis is used for RMSE, where a lower value indicates better performance, in contrast to the metrics $R^2$, Kendall's $\tau$, and Spearman's $\rho$.

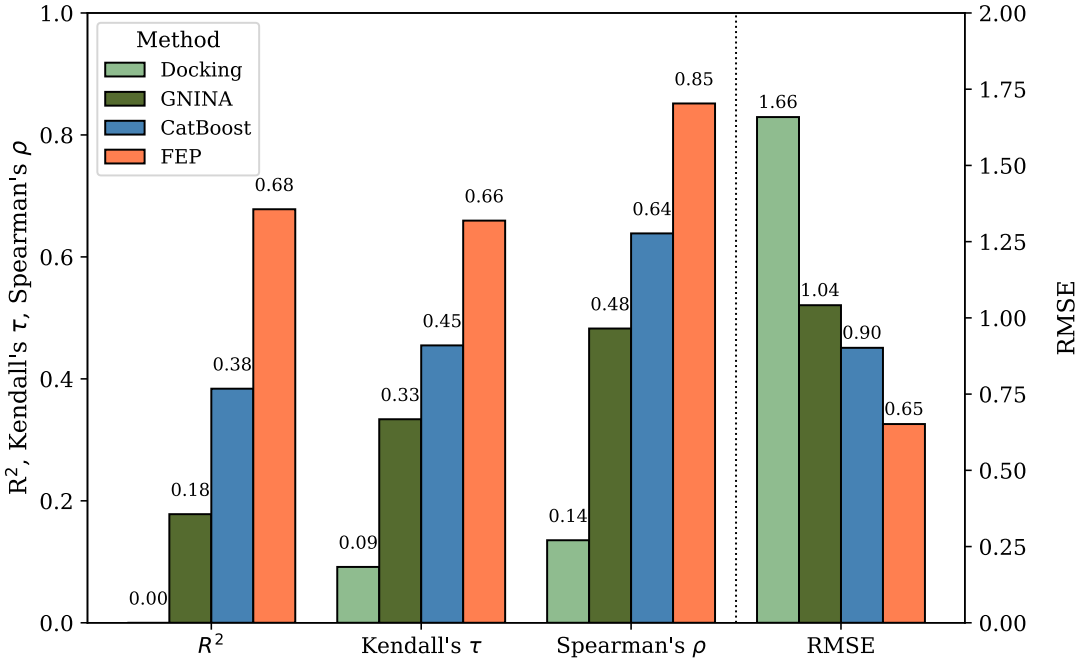



**Figure 4.4:** Final evaluation of model performance on the test set. The ML model is compared against Docking, GNINA, and FEP methods across multiple evaluation metrics. The left y-axis represents $R^2$, Kendall's $\tau$, and Spearman's $\rho$, while the right y-axis corresponds to RMSE.

The next evaluation on the test dataset assesses model performance per protein target, as shown in Figures 4.5–4.7. These figures illustrate the performance of the methods, including Docking, GNINA, CatBoost, and FEP, across various proteins. The performance is evaluated using Spearman's $\rho$, Kendall's $\tau$, and RMSE.

The ranking performance of the different methods is shown in Figure 4.5 and Figure 4.6. Overall, we observe that FEP outperforms the other methods across the various protein targets for both Spearman's $\rho$ and Kendall's $\tau$. There are a few cases where the CatBoost model performs best, specifically, on the protein targets *cdk2* and *syk* when considering Spearman's $\rho$, and on *cdk2* and *cdk8* in when considering Kendall's $\tau$. Interestingly, GNINA outperforms the CatBoost model in some cases, which is worth noting.

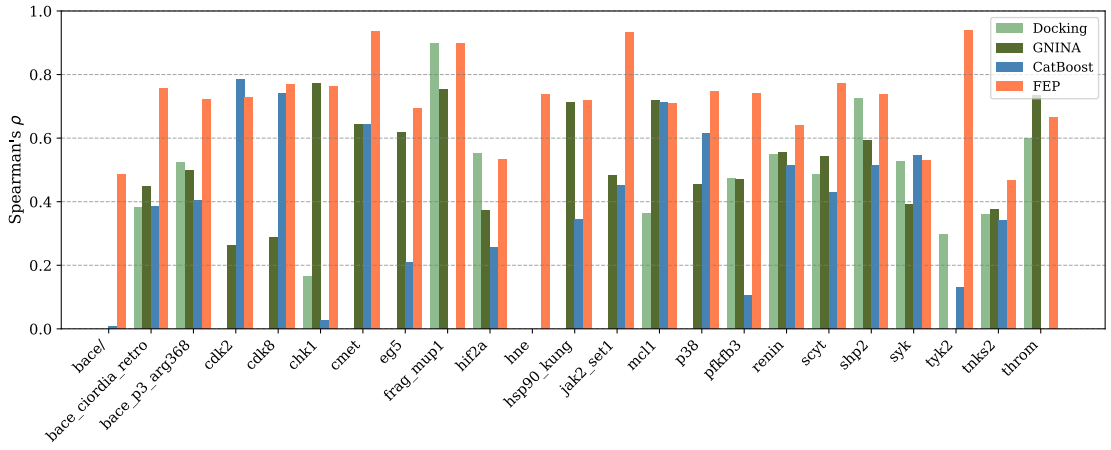



**Figure 4.5:** Comparison of Spearman's $\rho$ values for model performance across various protein targets.

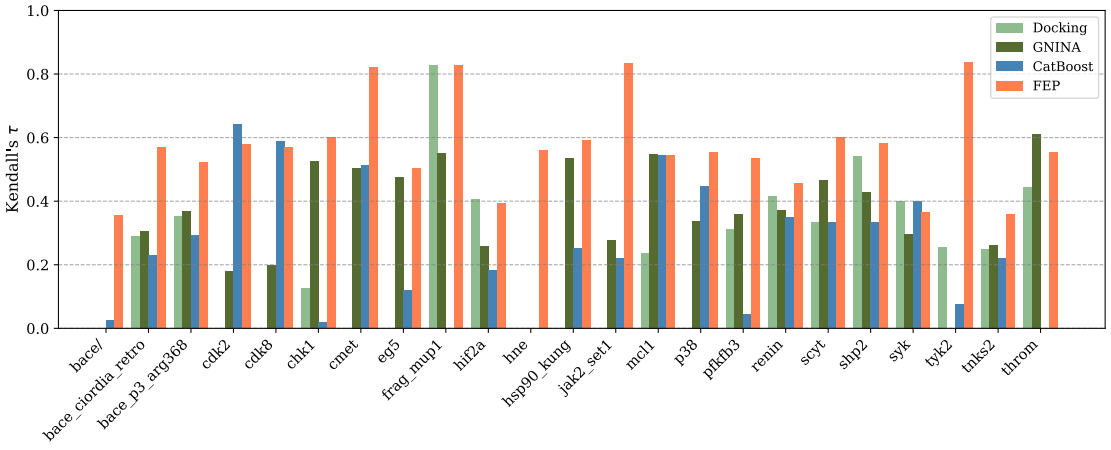



**Figure 4.6:** Comparison of Kendall's $\tau$ values for model performance across various protein targets.

The RMSE metric, shown in Figure 4.7, displays a similar trend, with FEP outperforming the other methods for 18 out of 23 protein targets. The CatBoost model performs best on four proteins, while GNINA performs best on only one protein, *tnks2*. In several cases, such as *bace, bace_ciordia_retro, chk1, eg5*, and *hif2a*, GNINA outperforms the ML model.

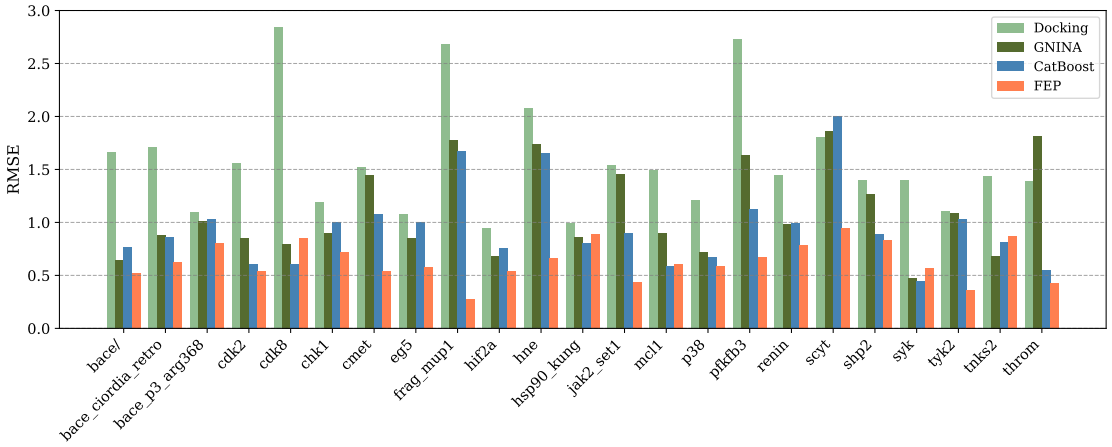



**Figure 4.7:** Comparison of RMSE values for model performance across various protein targets.

# 5

# Discussion

## 5.1 Model Performance and Selection

The creation of the baseline models demonstrated that using CNNaffinity as the input feature to a linear regression model resulted in better performance than using minimizedAffinity. This outcome is expected, as CNNaffinity represents an enhanced binding affinity prediction generated by the GNINA software, which utilizes a CNN to re-score docking poses and is thus more likely to correlate with the target values.

An interesting and somewhat unexpected finding was that using minimizedAffinity as input and applying a similarity-based split led to better performance than using a random split. Typically, random splits introduce more data leakage, which would be expected to inflate performance. One possible explanation is that minimizedAffinity does not correlate well with the target values. As a result, the model can not take advantage of the leaked data.

Several ML models were evaluated when selecting an ML model for binding affinity prediction. The best-performing models were tree-based models, which is expected, as these models can capture complex, non-linear relationships in the data, something linear models fail to do. The top-performing models showed similar results, suggesting that further evaluating and tuning additional candidates could have been beneficial in determining whether any could outperform the CatBoost model.

As anticipated, model performance declined when using a random split compared to a similarity-based split. This emphasizes the importance of addressing potential data leakage for a more realistic performance measure. Comparing the ML models to the baseline linear regression model, it was clear that incorporating all available features, MD-derived, docking features, and Ligand Properties, and using more complex models improved performance. This suggests that MD features may be valuable for estimating binding affinity.

CatBoost, the best-performing model, was further optimized through hyperparameter tuning, resulting in a slight improvement in the validation set. The relatively small performance gain indicates that the model performed well without tuning. Hyperparameter tuning can lead to a model specific to the training data, potentially reducing the model's ability to generalize. Therefore, given the strong performance

of the untuned model, it may be advantageous to explore how these generalize to unseen data.

## 5.2 Feature Set Evaluation

Table 4.5 presents the performance of the evaluated feature sets. The MD features yielded better predictive performance when using MD features alone versus Docking Features alone. This suggests that the entropic contributions to the binding, captured by MD features, play a significant role in predicting binding affinity, a factor that docking approaches do not effectively account for. Surprisingly, combining MD features with Ligand Properties resulted in performance comparable to combining MD and Docking Features. One might expect Docking Features to add more value than the relatively simple Ligand Properties, but this was not the case. An explanation for this is that MD still captures many of the properties that Docking captures (e.g., enthalpic contributions), making the Docking features somewhat redundant. This further reinforces the importance of incorporating MD features when predicting binding affinity. The best-performing feature set was refined using correlation-based feature selection, where Spearman correlation was used to remove redundant and highly correlated features. This reduction in feature dimensionality, without sacrificing performance, highlights the value of feature selection and motivates further investigation into which specific features contribute most to prediction accuracy.

## 5.3 Final Model Performance

One of the aims of this thesis was to understand how molecular properties influence binding affinity. The results in Table 4.6 clearly show that the final ML model relied most heavily on the MD features and Ligand Properties, with Docking Features contributing significantly less to the predictions. This again underscores the importance of incorporating MD features when predicting binding affinity, but surprisingly enough, something as easy to extract as Ligand Properties is a good addition to the prediction of binding affinity. Dihedral Angles, RMSF, and SASA contributed the most to the model's predictions among the MD features. This is expected, as these features are more closely associated with entropy, whereas the others are primarily related to enthalpy. Interestingly, CNNaffinity did not appear as one of the most important features despite the initial assumption that it would correlate strongly with the target values.

As shown in Figure 4.3, the method developed in this thesis outperforms standard docking approaches for predicting binding affinity. This highlights the potential of the proposed method to significantly improve binding affinity prediction.

The final evaluation comparing the ML approach with traditional Docking, GNINA,

and FEP binding affinity predictions showed that the CatBoost model outperformed standard Docking and GNINA but was outperformed by FEP. The most surprising aspect of this result was the noticeable drop in performance from the validation set to the test set. Potential explanations for this decline include a poor generalization of the model to the new dataset or a significant difference in PL complexes between the training and test sets, such that the model encountered entirely unfamiliar examples during testing. The performance drop is most likely due to the differences in ligands: the test set consisted of a congeneric series of ligands for each protein, whereas the training set included a broader and more diverse set of ligands. As a result, the model may have learned to distinguish between strong binders and weak binders in general but struggled to capture significant binding affinity changes that occur when only minor modifications are made to a ligand. Based on this understanding, the approach considered in this thesis would be more effective during the hit-finding stage of drug discovery compared to the lead optimization step (the training set was more similar to the hit-finding, and the test set was more similar to the lead optimization).

Furthermore, as described in the dataset section, the training and validation sets consisted of PL complexes derived from X-ray crystal structures, whereas the test set structures were generated via molecular Docking. This structural difference may have negatively impacted model performance, particularly since the MD features used during training were generated from simulations starting from experimentally determined structures. In contrast, the test set simulations were based on docked poses, potentially leading to different dynamic behaviors and MD-derived features that differ from those seen in the training data.

Another unexpected finding appeared when analyzing the performance of the different methods on individual proteins. In many cases, GNINA outperformed the CatBoost model. This is surprising because the CatBoost model was trained using GNINA's predicted affinity (CNNaffinity) as one of its input features, which would be expected to enhance the CatBoost model's predictive capability. Again, a likely explanation lies in the differences between the PL complexes in the training and test sets, as well as the differences in the starting structures. The test set structures were generated via Docking, whereas the training set structures were based on X-ray crystallography. As a result, docking-derived features, such as CNNaffinity, may have been less reliable or consistent in the test set.

## 5.4 Limitations and Future work

The primary limitation of this study was the restricted time frame, which constrained the level of detail with which certain aspects could be addressed. One key limitation was the exclusive use of an internally generated dataset from AstraZeneca without incorporating publicly available datasets such as those described by Liu et al. [3]. Including publicly available datasets could have introduced greater diver-

sity in PL complexes, potentially resulting in a model with improved generalization across different types of complexes. The training dataset was not analyzed in detail; it was treated more as a diverse set of protein targets and ligand series. Future work could involve evaluating model performance across specific classes of protein targets to improve predictions for underperforming groups.

Another limitation was the reliance on standard MD simulation settings without exploring alternative simulation types or parameter variations. The quality of the features derived from simulations can be highly sensitive to simulation protocols. Allocating more time to the simulations could have led to higher-quality features and, thus, better model predictions. The theoretical background describes that the Gibbs free energy determines binding affinity, the energy difference between the unbound state (protein and ligand separated in water) and the bound state (PL complex in water). Running additional short simulations of the unbound state could offer valuable insights into binding affinity and improve the predictive features.

Additionally, as stated in the Methods section, water molecules were removed from the MD trajectories to save disk space. This decision meant that all features, except interaction energies, were extracted without considering solvent interactions, which may have affected the quality of the derived features. Future studies could investigate whether retaining water in the trajectories improves the model's predictive performance. Since the features used in this thesis already yielded promising results, no further exploration into feature engineering was conducted. However, future work could build on the feature importance findings to extract features that more accurately represent binding affinity. For example, dihedral angles, RMSF, and SASA contributed significantly to model performance, and more refined representations of these features could be explored.

Another potential direction is to make feature extraction more data-driven by providing the whole MD trajectory as input to an ML model. This would allow the model to learn relevant patterns and extract important features automatically for direct binding affinity prediction.

In this thesis, only a few regression models were evaluated, most of which were linear or tree-based. Neural networks were also tested, but they did not perform as well as the tree-based models, primarily because they overfitted quickly and were not explored further due to time constraints. For future work, it would be interesting to investigate whether neural networks could improve predictive performance when applied to larger datasets. Exploring alternative approaches, such as ensemble methods that combine multiple models, could enhance prediction accuracy.

Since the test set consisted of different proteins, each with a congeneric ligand series, it would have been beneficial for the training data to include similarly structured examples rather than just a diverse set of ligands. This would allow the model to learn that binding affinity can change significantly even when the ligand structure undergoes only minor modifications. The importance of this is illustrated in Figure 4.4, where the model performed well overall on the entire test set, but the perfor-

mance varied notably when evaluated on individual proteins, Figure 4.5-4.7. This suggests that the model struggles to capture subtle ligand variations that lead to significant changes in binding affinity.

# 6

# Conclusion

This thesis focused on developing an ML model to predict PL binding affinities with accuracies approaching FEP methods. The ML model, which integrates features from MD simulations, docking, and ligand properties, demonstrated improved predictive performance over traditional docking techniques. The findings highlight the important role of MD-derived features, particularly in capturing entropic contributions to binding affinities.

Using a tree-based model, such as CatBoost, helped manage the complex relationships within the data. However, limitations were noted in the model's generalization, particularly due to differences between the test set (a congeneric series of ligands) and the training set. Features such as dihedral angles, RMSF, SASA, and ligand properties were instrumental in the model's performance. Although the developed method outperformed standard docking, it did not surpass FEP predictions, indicating areas for potential improvement.

Future work should focus on incorporating more diverse datasets, particularly those involving congeneric series of ligands, and exploring alternative MD protocols to enhance feature quality. Additionally, retaining solvent interactions during feature extraction could lead to more accurate predictions. Further exploration of feature extraction and ML approaches is recommended, particularly with larger datasets.

Overall, this study offers valuable insights into integrating MD-derived features for predicting PL binding, laying a solid foundation for continued advancements in the early stages of drug discovery.

# Bibliography


[1] P. Corr and D. Williams, *The pathway from idea to regulatory approval: Examples for drug development*, 2009.

[2] V. A. Adediwura, K. Koirala, H. N. Do, J. Wang, and Y. Miao, "Understanding the impact of binding free energy and kinetics calculations in modern drug discovery," *Expert Opinion on Drug Discovery*, pp. 1–12, 2024.

[3] X. Liu, S. Jiang, X. Duan, *et al.*, "Binding affinity prediction: From conventional to machine learning-based approaches," *arXiv preprint arXiv:2410.00709*, 2024.

[4] X. Du, Y. Li, Y.-L. Xia, *et al.*, "Insights into protein–ligand interactions: Mechanisms, models, and methods," *International journal of molecular sciences*, vol. 17, no. 2, p. 144, 2016.

[5] L. Di, E. H. Kerns, and G. T. Carter, "Drug-like property concepts in pharmaceutical design," *Current pharmaceutical design*, vol. 15, no. 19, pp. 2184–2194, 2009.

[6] S. Genheden and U. Ryde, "The mm/pbsa and mm/gbsa methods to estimate ligand-binding affinities," *Expert opinion on drug discovery*, vol. 10, no. 5, pp. 449–461, 2015.

[7] T. Pantsar and A. Poso, "Binding affinity via docking: Fact and fiction," *Molecules*, vol. 23, no. 8, p. 1899, 2018.

[8] W. J. Lima Silva and R. Ferreira de Freitas, "Assessing the performance of docking, fep, and mm/gbsa methods on a series of klk6 inhibitors," *Journal of Computer-Aided Molecular Design*, vol. 37, no. 9, pp. 407–418, 2023.

[9] D. L. Mobley and P. V. Klimovich, "Perspective: Alchemical free energy calculations for drug discovery," *The Journal of chemical physics*, vol. 137, no. 23, 2012.

[10] M. K. Gilson and H.-X. Zhou, "Calculation of protein-ligand binding affinities," *Annu. Rev. Biophys. Biomol. Struct.*, vol. 36, no. 1, pp. 21–42, 2007.

[11] Z. Cournia, B. Allen, and W. Sherman, "Relative binding free energy calculations in drug discovery: Recent advances and practical considerations," *Journal of chemical information and modeling*, vol. 57, no. 12, pp. 2911–2937, 2017.

[12] D. S. Spassov, "Binding affinity determination in drug design: Insights from lock and key, induced fit, conformational selection, and inhibitor trapping models," *International Journal of Molecular Sciences*, vol. 25, no. 13, p. 7124, 2024.

[13] R. Dias and B. Kolaczkowski, "Improving the accuracy of high-throughput protein-protein affinity prediction may require better training data," *BMC bioinformatics*, vol. 18, pp. 7–18, 2017.

[14] J. Ma, "Usefulness and limitations of normal mode analysis in modeling dynamics of biomolecular complexes," *Structure*, vol. 13, no. 3, pp. 373–380, 2005.

[15] A. T. McNutt, P. Francoeur, R. Aggarwal, *et al.*, "Gnina 1.0: Molecular docking with deep learning," *Journal of cheminformatics*, vol. 13, no. 1, p. 43, 2021.

[16] O. Trott and A. J. Olson, "Autodock vina: Improving the speed and accuracy of docking with a new scoring function, efficient optimization, and multithreading," *Journal of computational chemistry*, vol. 31, no. 2, pp. 455–461, 2010.

[17] D. R. Koes, M. P. Baumgartner, and C. J. Camacho, "Lessons learned in empirical scoring with smina from the csar 2011 benchmarking exercise," *Journal of chemical information and modeling*, vol. 53, no. 8, pp. 1893–1904, 2013.

[18] P. Saudagar and T. Tripathi, *Protein folding dynamics and stability: experimental and computational methods*. Springer Nature, 2023.

[19] S. Shechter, R. K. Pal, F. Trovato, O. Rozen, M. J. Gage, and D. Avni, "P70s6k as a potential anti-covid-19 target: Insights from wet bench and in silico studies," *Cells*, vol. 13, no. 21, p. 1760, 2024.

[20] S. A. Hollingsworth and R. O. Dror, "Molecular dynamics simulation for all," *Neuron*, vol. 99, no. 6, pp. 1129–1143, 2018.

[21] D. Frenkel and B. Smit, *Understanding molecular simulation: from algorithms to applications*. Elsevier, 2023.

[22] C.-E. A. Chang, Y.-M. M. Huang, L. J. Mueller, and W. You, "Investigation of structural dynamics of enzymes and protonation states of substrates using computational tools," *Catalysts (Basel, Switzerland)*, vol. 6, no. 6, p. 82, 2016.

[23] F. A. Bais and J. D. Farmer, "The physics of information," *arXiv preprint arXiv:0708.2837*, 2007.

[24] D. R. Houston and M. D. Walkinshaw, "Consensus docking: Improving the reliability of docking in a virtual screening context," *Journal of chemical information and modeling*, vol. 53, no. 2, pp. 384–390, 2013.

[25] J. Jumper, R. Evans, A. Pritzel, *et al.*, "Highly accurate protein structure prediction with alphafold," *nature*, vol. 596, no. 7873, pp. 583–589, 2021.

[26] J. Lemkul, "From proteins to perturbed hamiltonians: A suite of tutorials for the gromacs-2018 molecular simulation package," *Living J Comput Mol Sci*, vol. 1, no. 1, 1â, 2019.

[27] S. De Vita, M. G. Chini, G. Bifulco, and G. Lauro, "Insights into the ligand binding to bromodomain-containing protein 9 (brd9): A guide to the selection of potential binders by computational methods," *Molecules*, vol. 26, no. 23, p. 7192, 2021.

[28] M. L. Verteramo, O. Stenstrom, M. M. Ignjatovic, *et al.*, "Interplay between conformational entropy and solvation entropy in protein–ligand binding," *Journal of the American Chemical Society*, vol. 141, no. 5, pp. 2012–2026, 2019.

[29] C. Cao, L. Wang, X. Chen, S. Zou, G. Wang, and S. Xu, "Amino acids in nine ligand-prefer ramachandran regions," *BioMed Research International*, vol. 2015, no. 1, p. 757 495, 2015.

[30] A. Kumar and K. K. Ojha, "Molecular dynamics simulation methods to study structural dynamics of proteins," in *Protein Folding Dynamics and Stability: Experimental and Computational Methods*, Springer, 2023, pp. 83–106.

[31] J. Wang and T. Hou, "Develop and test a solvent accessible surface area-based model in conformational entropy calculations," *Journal of chemical information and modeling*, vol. 52, no. 5, pp. 1199–1212, 2012.

[32] S. J. Grabowski, *Hydrogen bonding: new insights*. Springer, 2006, vol. 3.

[33] *Hydrogen bond*, `https://www.sciencedirect.com/topics/biochemistry-genetics-and-molecular-biology/hydrogen-bond`, [Online]. Available: `https://www.sciencedirect.com/topics/biochemistry-genetics-and-molecular-biology/hydrogen-bond` [Accessed: Apr. 23, 2025], ScienceDirect, n.d.

[34] P. Mohanty, R. Agrata, B. I. Habibullah, A. GS, and R. Das, "Deamidation disrupts native and transient contacts to weaken the interaction between ubc13 and ring-finger e3 ligases," *Elife*, vol. 8, e49223, 2019.

[35] C. J. Camacho and S. Vajda, "Protein docking along smooth association pathways," *Proceedings of the National Academy of Sciences*, vol. 98, no. 19, pp. 10 636–10 641, 2001.

[36] Y. Kawasaki and E. Freire, "Finding a better path to drug selectivity," *Drug discovery today*, vol. 16, no. 21-22, pp. 985–990, 2011.

[37] C. M. Bishop and N. M. Nasrabadi, *Pattern recognition and machine learning*. Springer, 2006, vol. 4.

[38] A. Lindholm, N. Wahlström, F. Lindsten, and T. B. Schön, *Machine learning: a first course for engineers and scientists*. Cambridge University Press, 2022.

[39] T. Chen and C. Guestrin, "Xgboost: A scalable tree boosting system," in *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*, 2016, pp. 785–794.

[40] G. Ke, Q. Meng, T. Finley, *et al.*, "Lightgbm: A highly efficient gradient boosting decision tree," *Advances in neural information processing systems*, vol. 30, 2017.

[41] A. V. Dorogush, V. Ershov, and A. Gulin, "Catboost: Gradient boosting with categorical features support," *arXiv preprint arXiv:1810.11363*, 2018.

[42] S. Kaufman, S. Rosset, C. Perlich, and O. Stitelman, "Leakage in data mining: Formulation, detection, and avoidance," *ACM Transactions on Knowledge Discovery from Data (TKDD)*, vol. 6, no. 4, pp. 1–21, 2012.

[43] S. Kapoor and A. Narayanan, "Leakage and the reproducibility crisis in machine-learning-based science," *Patterns*, vol. 4, no. 9, 2023.

[44] D. Berrar *et al.*, *Cross-validation.* 2019.

[45] P. I. Frazier, "A tutorial on bayesian optimization," *arXiv preprint arXiv:1807.02811*, 2018.

[46] C. Xiao, J. Ye, R. M. Esteves, and C. Rong, "Using spearman's correlation coefficients for exploratory data analysis on big dataset," *Concurrency and Computation: Practice and Experience*, vol. 28, no. 14, pp. 3866–3878, 2016.

[47] R. E. Walpole, R. H. Myers, *et al.*, *Probability and statistics for engineers and scientists*. Prentice Hall, 2009.

[48] S. Khalid, T. Khalil, and S. Nasreen, "A survey of feature selection and feature extraction techniques in machine learning," in *2014 science and information conference*, IEEE, 2014, pp. 372–378.

[49] Schrödinger, LLC, *FEP+ Software*, `https://www.schrodinger.com/fep`, Accessed June 2025, n.d.

[50] D. Bajusz, A. Rácz, and K. Héberger, "Why is tanimoto index an appropriate choice for fingerprint-based similarity calculations?" *Journal of cheminformatics*, vol. 7, pp. 1–13, 2015.

[51] M. Ragoza, J. Hochuli, E. Idrobo, J. Sunseri, and D. R. Koes, "Protein–ligand scoring with convolutional neural networks," *Journal of chemical information and modeling*, vol. 57, no. 4, pp. 942–957, 2017.

[52] X. Daura, K. Gademann, B. Jaun, D. Seebach, W. F. van Gunsteren, and A. E. Mark, "Peptide folding: When simulation meets experiment," *Angewandte Chemie International Edition*, vol. 38, pp. 236–240, 1999. DOI: `10.1002/(SICI)1521-3773(19990115)38:1/2<236::AID-ANIE236>3.0.CO;2-M`.

[53] B. Knapp, L. Ospina, and C. M. Deane, "Avoiding false positive conclusions in molecular simulation: The importance of replicas," *Journal of Chemical Theory and Computation*, vol. 14, no. 12, pp. 6127–6138, 2018.

[54] N. Michaud-Agrawal, E. J. Denning, T. B. Woolf, and O. Beckstein, "Mdanalysis: A toolkit for the analysis of molecular dynamics simulations. 32 (10): 2319–2327," *URL: https://onlinelibrary. wiley. com/doi/abs/10.1002/jcc*, vol. 21787,

[55] M. J. Abraham, T. Murtola, R. Schulz, *et al.*, "Gromacs: High performance molecular simulations through multi-level parallelism from laptops to supercomputers," *SoftwareX*, vol. 1, pp. 19–25, 2015.

[56] G. Landrum, "Rdkit: Open-source cheminformatics http://www. rdkit. org," *Google Scholar There is no corresponding record for this reference*, vol. 3, no. 8, 2016.

[57] F. Boyles, C. M. Deane, and G. M. Morris, "Learning from the ligand: Using ligand-based features to improve binding affinity prediction," *Bioinformatics*, vol. 36, no. 3, pp. 758–764, 2020.

[58] A. H. Victoria and G. Maragatham, "Automatic tuning of hyperparameters using bayesian optimization," *Evolving Systems*, vol. 12, no. 1, pp. 217–223, 2021.

# A

# Appendix 1

## A.1 RDKit Ligand Descriptors

This section lists the molecular descriptors used as features in the model. These descriptors were computed using RDKit.

**Table A.1:** Summary of RDKit-derived ligand descriptors and their interpretations.

| **Feature Name** | **Description** |
|---|---|
| Molecular Weight | Total molecular weight of the compound |
| LogP | Octanol-water partition coefficient |
| TPSA | Topological polar surface area |
| Num Rotatable Bonds | Number of rotatable bonds |
| Num H Donors | Number of hydrogen bond donors |
| Num H Acceptors | Number of hydrogen bond acceptors |
| Num Rings | Number of rings |
| Heavy Atom Count | Number of non-hydrogen atoms |
| Fraction sp3 | Fraction of sp3 hybridized carbon atoms |
| Max Partial Charge | Maximum Gasteiger partial charge |
| Min Partial Charge | Minimum Gasteiger partial charge |
| Balaban J | Balaban J topological index |
| Molar Refractivity | Molar refractivity |
| Num Aromatic Rings | Number of aromatic rings |
| Num Aliphatic Rings | Number of aliphatic rings |
| Num Chiral Centers | Number of chiral centers |
| Fraction Aromatic Atoms | Ratio of aromatic atoms to heavy atoms |
| Chi0v | Valence molecular connectivity index (order 0) |
| Kappa1 | Kier's first kappa shape index |
| Num Heteroatoms | Number of heteroatoms |
| Num 5-Membered Rings | Number of 5-membered rings |
| Num 6-Membered Rings | Number of 6-membered rings |
| Exact Molecular Weight | Precise molecular weight including isotopes |
| Num Valence Electrons | Total number of valence electrons |
| Chi1v | Valence molecular connectivity index (order 1) |
| Kappa2 | Kier's second kappa shape index |
| Kappa3 | Kier's third kappa shape index |
| Num Spiro Atoms | Number of spiro atoms |
| Num Bridgehead Atoms | Number of bridgehead atoms |
| Num Lipinski Violations | Number of Lipinski rule violations |

| **Feature Name** | **Description** |
|---|---|
| Num Atoms | Total number of atoms |
| Num Sulfur Atoms | Count of sulfur atoms |
| Num Oxygen Atoms | Count of oxygen atoms |
| Num Nitrogen Atoms | Count of nitrogen atoms |
| Bertz CT | Bertz complexity index |
| Chi0n – Chi4n | Connectivity indices (order 0 to 4, non-valence) |
| Chi2v – Chi4v | Connectivity indices (order 2 to 4, valence) |
| HallKierAlpha | Hall–Kier alpha descriptor |
| Num Aliphatic Carbocycles | Count of aliphatic carbocyclic rings |
| Num Aliphatic Heterocycles | Count of aliphatic heterocyclic rings |
| Num Amide Bonds | Count of amide bonds |
| Num Aromatic Carbocycles | Count of aromatic carbocyclic rings |
| Num Aromatic Heterocycles | Count of aromatic heterocyclic rings |
| Num Atom Stereo Centers | Number of atomic stereocenters |
| Num Heterocycles | Count of heterocyclic rings |
| Num Saturated Carbocycles | Count of saturated carbocyclic rings |
| Num Saturated Heterocycles | Count of saturated heterocyclic rings |
| Num Saturated Rings | Total number of saturated rings |
| BCUT2D_Sum | Sum of BCUT2D eigenvalues |
| AUTOCORR2D_Sum | Sum of 2D autocorrelation values |
| Asphericity | Measure of shape anisotropy |
| Crippen_LogP | LogP from Crippen model |
| Crippen_MR | Molar refractivity from Crippen model |
| GETAWAY_Sum | Sum of GETAWAY descriptors |
| InertialShapeFactor | Inertial shape factor |
| LabuteASA | Approximate surface area (Labute) |
| MORSE_Sum | Sum of 3D-MoRSE descriptors |
| NPR1 | Normalized principal moment ratio 1 |
| NPR2 | Normalized principal moment ratio 2 |
| PBF | Plane of best fit |
| PMI1 – PMI3 | Principal moments of inertia (1 to 3) |
| Phi | Angle between principal axes |
| RDF_Sum | Sum of radial distribution function values |
| RadiusOfGyration | Radius of gyration |
| WHIM_Sum | Sum of WHIM descriptors |
| SlogP_VSA_Sum | Sum of SlogP–VSA values |
| SMR_VSA_Sum | Sum of SMR–VSA values |
| PEOE_VSA_Sum | Sum of PEOE–VSA values |
| MQNs_Sum | Sum of Molecular Quantum Numbers |

**DEPARTMENT OF PHYSICS**
**CHALMERS UNIVERSITY OF TECHNOLOGY**
Gothenburg, Sweden
www.chalmers.se


CHALMERS
UNIVERSITY OF TECHNOLOGY